NASA/CR-2001-211256
ICASE Report No. 2001-38

High Order WENO Schemes for Hamilton-Jacobi Equations on Triangular Meshes

Yong-Tao Zhang and Chi-Wang Shu
Brown University, Providence, Rhode Island

ICASE
NASA Langley Research Center
Hampton, Virginia

Operated by Universities Space Research Association

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

Prepared for Langley Research Center
under Contract NAS1-97046

December 2001

HIGH ORDER WENO SCHEMES FOR HAMILTON-JACOBI EQUATIONS ON TRIANGULAR MESHES *

YONG-TAO ZHANG† AND CHI-WANG SHU‡

Abstract. In this paper we construct high order weighted essentially non-oscillatory (WENO) schemes for solving the nonlinear Hamilton-Jacobi equations on two-dimensional unstructured meshes. The main ideas are nodal based approximations, the usage of monotone Hamiltonians as building blocks on unstructured meshes, nonlinear weights using smooth indicators of second and higher derivatives, and a strategy to choose diversified smaller stencils to make up the bigger stencil in the WENO procedure. Both third-order and fourth-order WENO schemes using combinations of second-order approximations with nonlinear weights are constructed. Extensive numerical experiments are performed to demonstrate the stability and accuracy of the methods. High-order accuracy in smooth regions, good resolution of derivative singularities, and convergence to viscosity solutions are observed.

Key words. weighted essentially non-oscillatory schemes, Hamilton-Jacobi equations, high-order accuracy, unstructured mesh

Subject classification. Applied and Numerical Mathematics

1. Introduction. In this paper, we consider the numerical solution of Hamilton-Jacobi (H-J) equations

$$\phi_t + H(\phi_{x_1}, \cdots, \phi_{x_d}) = 0, \qquad \phi(x,0) = \phi_0(x). \tag{1.1}$$

Such equations appear often in applications, such as in optimal control, differential games, image processing and computer vision, and geometric optics. With the popularity of level set methods [12] the necessity to have good algorithms to solve H-J equations becomes even more obvious.

There have been many papers in the literature developing numerical methods to solve H-J equations. In certain sense H-J equations are easier to solve than their conservation laws counterparts, because the solutions here are smoother: typical solutions are continuous with possibly discontinuous derivatives.

For structured meshes, finite difference methods similar to those developed for conservation laws could be easily designed. Thus we have the earlier second order ENO schemes of Osher and Sethian [12], higher order essentially non-oscillatory (ENO) schemes of Osher and Shu [13], higher order weighted ENO (WENO) schemes of Jiang and Peng [7], and central high resolution schemes of Lin and Tadmor [10], among many others.

However, for unstructured meshes there are conceptional difficulties in designing schemes such as finite volume schemes and finite element schemes, which are very useful for conservation laws. The problem arises because the H-J equations cannot be written in a "conservation form", suitable for integration by parts which is the backbone of finite volume and finite element methods. Thus algorithms, especially high order algorithms for H-J equations on unstructured meshes are relatively few. We have the very good first and second order finite volume type schemes of Abgrall [1] and ENO or limiter type high order version of Augoula

*Research supported by ARO grant DAAD19-00-1-0405, NSF grants DMS-9804985 and ECS-9906606, NASA Langley grant NCC1-01035 and Contract NAS1-97046 while the second author was in residence at ICASE, NASA Langley Research Center, Hampton, VA 23681-2199, and AFOSR grant F49620-99-1-0077.

†Division of Applied Mathematics, Brown University, Providence, RI 02912. E-mail: zyt@cfm.brown.edu

‡Division of Applied Mathematics, Brown University, Providence, RI 02912. E-mail: shu@cfm.brown.edu

and Abgrall [2]. The monotone Hamiltonians developed in [1] are used in this paper as building blocks. We also have the continuous finite element methods of Barth and Sethian [4], and the discontinuous Galerkin methods of Hu and Shu [5]. However, the formulation of the finite element methods in [5] actually uses the differentiated version of H-J, which is a system of conservation laws. It would be desirable in many situations to use directly H-J on an unstructured mesh and still obtain high order, non-oscillatory numerical results. The algorithms developed in this paper fulfill this purpose.

The WENO methodology adopted in this paper can be traced back to the earlier work for conservation laws started in [11] for a third order finite volume version in one space dimension and in [8] for third and fifth order finite difference WENO schemes in multi space dimensions with a general framework for the design of the smoothness indicators and nonlinear weights. The techniques most relevant to the current paper are the third and fourth order finite volume WENO schemes for 2D conservation laws in general triangulations [6] and the finite difference WENO schemes for Hamilton-Jacobi equations [7]. However, significant new components of the algorithms have been developed in this paper to deal with the additional difficulty caused by the "non-conservation" form of the H-J equations and unstructured meshes. These include nodal based approximations, the usage of monotone Hamiltonians as building blocks on unstructured meshes, nonlinear weights using smooth indicators of second and higher derivatives, and a strategy to choose diversified smaller stencils to make up the bigger stencil in the WENO procedure. Both third-order and fourth-order WENO schemes using combinations of second-order approximations with nonlinear weights are constructed. Extensive numerical experiments are performed to demonstrate the stability and accuracy of the methods. High-order accuracy in smooth regions, good resolution of derivative singularities, and convergence to viscosity solutions are observed.

The algorithm is developed in section 2. Section 3 contains numerical examples verifying the stability, convergence and accuracy of the algorithms. Concluding remarks are given in section 4.

2. The WENO algorithm for 2D unstructured meshes. In this section we develop third and fourth order WENO schemes on unstructured meshes in 2D for the H-J equations.

2.1. The framework. We take $d = 2$ in (1.1), and use x, y instead of x_1, x_2:

$$\phi_t + H(\phi_x, \phi_y) = 0, \qquad \phi(x, y, 0) = \phi_0(x, y). \tag{2.1}$$

The equation (2.1) is solved in the domain Ω, which has a triangulation $\mathcal{T}_h$ consisting of triangles. The nodes will be named by their indices $0 \le i \le N$, with a total of $N+1$ nodes. For every node i, we define the $k_i + 1$ angular sectors $T_0, \cdots, T_{k_i}$ meeting at the point i; they are the inner angles at node i of the triangles having i as a vertex. The indexing of the angular sectors is ordered counterclockwise. $\vec{n}_{l+\frac{1}{2}}$ is the unit vector of the half-line $D_{l+\frac{1}{2}} = T_l \bigcap T_{l+1}$, and θ_l is the inner angle of sector T_l, $0 \le l \le k_i$; see Figure 2.1.

We will denote by ϕ_i the numerical approximation to the viscosity solution of (2.1) at node i. $(\nabla\phi)_0, \cdots, (\nabla\phi)_{k_i}$ will respectively represent the numerical approximation of $\nabla\phi$ at node i in each angular sector $T_0, \cdots, T_{k_i}$.

An important building block for the algorithms in this paper is the Lax-Friedrichs type monotone Hamiltonian for arbitrary triangulations developed by Abgrall in [1], which is a generalization of the Lax-Friedrichs monotone Hamiltonian for Cartesian meshes in Osher and Shu [13]. This monotone Hamiltonian is given by

$$\hat{H}((\nabla\phi)_0, \cdots, (\nabla\phi)_{k_i}) = H\left(\frac{\sum_{l=0}^{k_i} \theta_l (\nabla\phi)_l}{2\pi}\right) - \frac{\alpha}{\pi} \sum_{l=0}^{k_i} \beta_{l+\frac{1}{2}} \left(\frac{(\nabla\phi)_l + (\nabla\phi)_{l+1}}{2}\right) \cdot \vec{n}_{l+\frac{1}{2}} \tag{2.2}$$

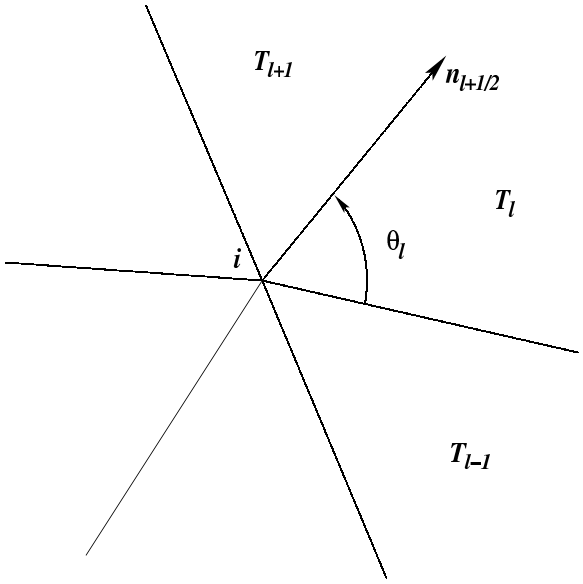

FIG. 2.1. *node i and its angular sectors.*

where

$$\beta_{l+\frac{1}{2}} = \tan\left(\frac{\theta_l}{2}\right) + \tan\left(\frac{\theta_{l+1}}{2}\right), \qquad \alpha = \max\{\max_{\substack{A\le u\le B\\ C\le v\le D}} |H_1(u,v)|, \max_{\substack{A\le u\le B\\ C\le v\le D}} |H_2(u,v)|\}.$$

Here H_1 and H_2 are the partial derivatives of H with respect to ϕ_x and ϕ_y, respectively, or the Lipschitz constants of H with respect to ϕ_x and ϕ_y, if H is not differentiable. $[A,B]$ is the value range for $(\phi_x)_l$, and $[C,D]$ is the value range for $(\phi_y)_l$, over $0 \le l \le k_i$ for the local Lax-Friedrichs Hamiltonian, and over $0 \le l \le k_i$ and $0 \le i \le N$ for global Lax-Friedrichs Hamiltonian.

The $\hat{H}$ in (2.2) defines a monotone Hamiltonian. It is Lipschitz continuous in all arguments and is consistent with H, i.e., $\hat{H}(\nabla\phi, \cdots, \nabla\phi) = H(\nabla\phi)$. Hence if we have high-order approximations to $\nabla\phi$ at node i in every angular sector, the numerical Hamiltonian $\hat{H}$ will be a high-order approximation to H.

The semi-discrete scheme is thus given by:

$$\frac{d}{dt}\phi_i(t) + \hat{H}((\nabla\phi)_0, \cdots, (\nabla\phi)_{k_i}) = 0 \tag{2.3}$$

and it is discretized in time by the high order nonlinearly stable Runge-Kutta time discretization in [16]. If the spatial dimension reduces to one, the numerical Hamiltonian (2.2) will reduce to the local or global Lax-Friedrichs Hamiltonian [13]:

$$\hat{H}(u^-, u^+) = H\left(\frac{u^- + u^+}{2}\right) - \frac{1}{2}\alpha(u^+ - u^-) \tag{2.4}$$

with $\alpha = \max_{A\le u\le B} |H'(u)|$. If the maximum for computing α is taken over the range covered by u^- and u^+, we get the local Lax-Friedrichs Hamiltonian; if it is also taken over all grid points, we get the global Lax-Friedrichs Hamiltonian.

2.2. Linear schemes. Now we discuss how to construct a high-order approximation for $\nabla\phi$ in every angular sector of every node. Let P^k denote the set of two-dimensional polynomials of degree less than or equal to k. We use Lagrange interpolations as follows: given a smooth function ϕ, and a triangulation with

triangles $\{\triangle_0, \triangle_1, \dots, \triangle_M\}$ and nodes $\{0, 1, 2, \dots, N\}$, we would like to construct, for each triangle $\triangle_i$, a polynomial $p(x, y) \in P^k$, such that $p(x_l, y_l) = \phi(x_l, y_l)$, where (x_l, y_l) are the coordinates of the three nodes of the triangle $\triangle_i$ and a few neighboring nodes. $p(x, y)$ would thus be a $(k+1)$th-order approximation to ϕ on the cell $\triangle_i$.

Because kth degree polynomial $p(x, y)$ has $K = \frac{(k+1)(k+2)}{2}$ degrees of freedom, we need to use the information of at least K nodes. In addition to the three nodes of the triangle $\triangle_i$, we may take the other $K-3$ nodes of the neighboring cells around triangle $\triangle_i$. We rename these K nodes as $S_i = \{M_1, M_2, \dots, M_K\}$, S_i is called a big stencil for the triangle $\triangle_i$. Let (x_i, y_i) be the barycenter of $\triangle_i$. Define $\xi = (x - x_i)/h_i$, $\eta = (y - y_i)/h_i$, where $h_i = \sqrt{|\triangle_i|}$ with $|\triangle_i|$ denoting the area of the triangle $\triangle_i$, then we can write $p(x, y)$ as:

$$p(x, y) = \sum_{j=0}^{k} \sum_{s+r=j} a_{sr} \xi^s \eta^r.$$

Using the K interpolation conditions:

$$p(M_l) = \phi(M_l), \qquad l = 1, 2, \cdots, K,$$

we get a $K \times K$ linear system for the K unknowns a_{sr}. The normalized variables ξ, η are used to make the condition number of the linear system independent of the mesh size.

It is well known that in two and higher dimensions this interpolation problem is not always well defined. The linear system can be very ill-conditioned or even singular, in such cases we would have to add more nodes to the big stencil S_i from the neighboring cells around triangle $\triangle_i$ to obtain an over-determined linear system, and then use the least-square method to solve it.

After we have obtained the approximation polynomial $p(x, y)$ on the triangle $\triangle_i$, ∇p will be a kth-order approximation for $\nabla\phi$ on $\triangle_i$. Hence we get the high-order approximation $\nabla p(x_l, y_l)$ to $\nabla\phi(x_l, y_l)$, for any one of the three vertices (x_l, y_l) of the triangle $\triangle_i$, in the relevant angular sectors.

2.2.1. Third-order linear schemes. A scheme is called linear if it is linear when applied to a linear equation with constant coefficients. We need a third-order approximation for $\nabla\phi$ to construct a third-order linear scheme, hence we need a cubic polynomial interpolation. A cubic polynomial p^3 has 10 degrees of freedom. We will use some or all of the nodes shown in Figure 2.2 to form our big stencil. For our target triangle $\triangle_0$, which has three vertices i, j, k and the barycenter G, we need to construct a cubic polynomial p^3, then ∇p^3 will be a third-order approximation for $\nabla\phi$ on $\triangle_0$, and the values of ∇p^3 at points i, j and k will be third-order approximations for $\nabla\phi$ at the angular sector $\triangle_0$ of nodes i, j and k. We name the nodes of the neighboring triangles of triangle $\triangle_0$ as follows: nodes 1, 4, 7 are the nodes (other than i, j, k) of neighbors of $\triangle_0$, nodes 9, 2, 3, 5, 6, 8 (other than 1, 4, 7, i, j, k) are the nodes of the neighbors of the three neighboring triangles of $\triangle_0$. Notice that the points 9, 2, 3, 5, 6, 8 do not have to be six distinct points. For example the points 5 and 6 could be the same point.

Our interpolation points are nodes i, j, k, 1, 4, 7 and some of the nodes taken from the set $E = \{9, 2, 3, 5, 6, 8\}$. The algorithm to determine the big stencil S_0 for triangle $\triangle_0$ is as follows.

Procedure 2.1: The choice of the big stencil for the third-order scheme.

1. Nodes i, j, k, 1, 4, 7 are always included in S_0.
2. Compute the distance of point G and every node in the set $E = \{9, 2, 3, 5, 6, 8\}$ respectively. Sort the nodes in E with this distance from smallest to biggest, and denote the sorted node set by $\hat{E}$.

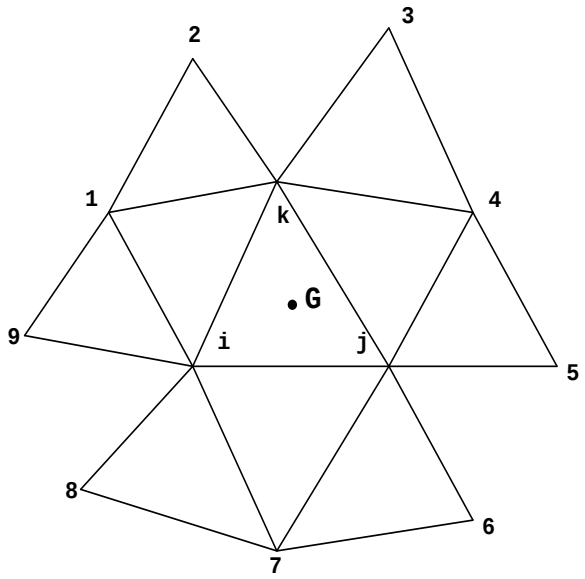

FIG. 2.2. *The nodes used for the big stencil of the third-order scheme.*

3. From the sorted node set $\hat{E}$, add the first 4 nodes to S_0. Use the 10 nodes in S_0 as interpolation points to get the 10×10 interpolation coefficient matrix A,

$$A = \begin{bmatrix} 1 & \xi_0 & \eta_0 & \xi_0^2 & \eta_0^2 & \xi_0\eta_0 & \xi_0^3 & \eta_0^3 & \xi_0^2\eta_0 & \xi_0\eta_0^2 \\ 1 & \xi_1 & \eta_1 & \xi_1^2 & \eta_1^2 & \xi_1\eta_1 & \xi_1^3 & \eta_1^3 & \xi_1^2\eta_1 & \xi_1\eta_1^2 \\ 1 & \xi_2 & \eta_2 & \xi_2^2 & \eta_2^2 & \xi_2\eta_2 & \xi_2^3 & \eta_2^3 & \xi_2^2\eta_2 & \xi_2\eta_2^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \xi_9 & \eta_9 & \xi_9^2 & \eta_9^2 & \xi_9\eta_9 & \xi_9^3 & \eta_9^3 & \xi_9^2\eta_9 & \xi_9\eta_9^2 \end{bmatrix}$$

where $\xi_l = (x_l - x_G)/h_G$, $\eta_l = (y_l - y_G)/h_G$, $h_G = \sqrt{|\triangle_0|}$, and (x_l, y_l) are the coordinates of the nodes in S_0, (x_G, y_G) is the coordinate of the point G.

4. Compute the reciprocal condition number c of A. This is provided by most linear solvers. If $c \geq \delta$ for some threshold δ, we have obtained the final stencil S_0. Otherwise, add the fifth node in $\hat{E}$ to S_0. Use the 11 nodes in S_0 as interpolation points to get the 11×10 least square interpolation coefficient matrix A. Judge the reciprocal condition number c again. Continue in doing this until $c \geq \delta$ is satisfied. In our computation, we have taken $\delta = 10^{-3}$ as a good threshold after extensive numerical experiments. Notice that, since we have normalized the coordinates, this threshold does not change when the mesh is scaled uniformly in all directions. For all the triangulations we have tested, at most 12 nodes are needed in S_0 to reach the condition $c \geq \delta$. ■

Remark 2.1. The reason that we always include the nodes i, j, k, 1, 4, 7 in S_0 is to make the target cell $\triangle_0$ sufficiently "central" in the stencil, thus avoiding S_0 to be seriously downwind biased which could lead to linear instability. Linear instability is indeed observed in our numerical experiments when 1, 4, 7 are not forced to be in S_0. ■

We now have obtained the big stencil S_0 and its associated cubic polynomial p^3. For each node (x_l, y_l) in $\triangle_0$, $\nabla p^3(x_l, y_l)$ is a third-order approximation to $\nabla\phi(x_l, y_l)$. In order to construct a high-order WENO scheme, an important step is to obtain a high-order approximation using a linear combination of lower order

approximations. We will use a linear combination of second-order approximations to get the same third-order approximation to $\nabla\phi(x_l, y_l)$ as $\nabla p^3(x_l, y_l)$, i.e., we require

$$\frac{\partial}{\partial x}p^3(x_l, y_l) = \sum_{s=1}^{q}\gamma_{s,x}\frac{\partial}{\partial x}p_s(x_l, y_l), \qquad \frac{\partial}{\partial y}p^3(x_l, y_l) = \sum_{s=1}^{q}\gamma_{s,y}\frac{\partial}{\partial y}p_s(x_l, y_l) \tag{2.5}$$

where p_s are quadratic interpolation polynomials, and $\gamma_{s,x}$ and $\gamma_{s,y}$ are the linear weights for the x-directional derivative and the y-directional derivative respectively, for $s = 1, \cdots, q$. The linear weights are constants depending only on the local geometry of the mesh. The equalities in (2.5) should hold for any choices of the function ϕ.

Notice that to get a second-order approximation for the derivatives $\nabla\phi(x_l, y_l)$, we need a quadratic interpolation polynomial. According to the argument in [6], the cubic polynomial $p^3(x, y)$ has four more degrees of freedom than each quadratic polynomial $p_s(x, y)$, namely x^3, x^2y, xy^2, y^3. For the six degrees of freedom $1, x, y, x^2, xy, y^2$, if we take $\phi = 1, \phi = x, \phi = y, \phi = x^2, \phi = xy$ and $\phi = y^2$, the equalities in (2.5) will hold for all these cases under only one constraint each on $\gamma_{s,x}$ and $\gamma_{s,y}$, namely $\sum_{s=1}^{q}\gamma_{s,x} = 1$ and $\sum_{s=1}^{q}\gamma_{s,y} = 1$, because p^3 and p_s all reproduce these functions exactly. Hence we should only need $q \geq 5$. We take $q = 5$ in our scheme.

We now need $q = 5$ small stencils $\Gamma_s, s = 1, \cdots, 5$ for the target triangle $\triangle_0$, satisfying $S_0 = \bigcup_{s=1}^{5}\Gamma_s$, and every quadratic polynomial p_s is associated with a small stencil Γ_s. In our third-order scheme, the small stencils will be the same for both directions x, y and all three nodes i, j, k in $\triangle_0$. However the linear weights $\gamma_{s,x}, \gamma_{s,y}$ can be different for different nodes i, j, k and different directions x, y. Because each quadratic polynomial has six degrees of freedom, the number of nodes in Γ_s must be at least six. To build a small stencil Γ_s, we start from several candidates $\Gamma_s^{(r)}, r = 1, 2, \cdots, n_s$. These candidates are constructed by first taking a point $A_s^{(r)}$ as the "center", then finding at least six nodes from S_0 which have the shortest distances from $A_s^{(r)}$ and can generate the interpolation coefficient matrix with a good condition number, using the method of Procedure 2.1. We then choose the best Γ_s among $\Gamma_s^{(r)}, r = 1, \cdots, n_s$ for every $s = 1, \cdots, 5$. Here "best" means that by using this group of small stencils, the linear weights $\gamma_{s,x}, \gamma_{s,y}, s = 1, \cdots, 5$ for all three nodes i, j, k are either all positive or have the smallest possible negative values in magnitude. The details of the algorithm is described in the following procedure.

Procedure 2.2: The third-order linear scheme.

For every triangle $\triangle_l, l = 1, \cdots, N$, do steps $1 \sim 5$:

1. Follow Procedure 2.1 to obtain the big stencil S_l for $\triangle_l$.
2. For $s = 1, \cdots, 5$, find the set $W_s = \{\Gamma_s^{(r)}, r = 1, 2, \cdots, n_s\}$, which are the candidate small stencils for the s-th small stencil. We use the following method to find the $\Gamma_s^{(r)}$ in W_s: first, nodes i, j, k are always included in every $\Gamma_s^{(r)}$; then we take a point $A_s^{(r)}$ as the center of $\Gamma_s^{(r)}$, detailed below, and using the idea of Procedure 2.1, we find at least 3 additional nodes other than i, j, k from S_l which satisfy the following two conditions: 1.) they have the shortest distances from $A_s^{(r)}$; and 2.) taking them and the nodes i, j, k as the interpolation points, we will obtain the interpolation coefficient matrix A with a good condition number, namely the reciprocal condition number c of A satisfies $c \geq \delta$ with the same threshold $\delta = 10^{-3}$. For the triangulations we have tested, at most 8 nodes are used to reach this threshold value. Finally, the center of the candidate stencils $A_s^{(r)}, r = 1, \cdots, n_s; s = 1, \cdots, 5$ are taken from the nodes around $\triangle_l$ (see Figure 2.2) as follows:
   - $A_1^{(1)} =$ point G, $n_1 = 1$;
   - $A_2^{(1)} =$ node 1, $A_2^{(2)} =$ node 2, $A_2^{(3)} =$ node 9, $n_2 = 3$;

- $A_3^{(1)}$ = node 4, $A_3^{(2)}$ = node 3, $A_3^{(3)}$ = node 5, $n_3 = 3$;
- $A_4^{(1)}$ = node 7, $A_4^{(2)}$ = node 6, $A_4^{(3)}$ = node 8, $n_4 = 3$;
- $\{A_5^{(r)}\}_{r=1}^{9}$ = nodes $2, 9, 3, 5, 6, 8$ and the middle points of nodes 2 and 3, 5 and 6, 9 and 8. $n_5 = 9$.

3. By taking one small stencil $\Gamma_s^{(r_s)}$ from each $W_s, s = 1, \cdots, 5$ to form a group, we obtain $n_1 \times n_2 \times \cdots \times n_5$ groups of small stencils. We eliminate the groups which contain the same small stencils, and also eliminate the groups which do not satisfy the condition

$$\bigcup_{s=1}^{5} \Gamma_s^{(r_s)} = S_l$$

According to every group $\{\Gamma_s^{(r_s)}, s = 1, \cdots, 5\}$ of small stencils, we have 5 quadratic polynomials $\{p_s^{(r_s)}\}_{s=1}^{5}$. We evaluate $\frac{\partial}{\partial x} p_s^{(r_s)}$ and $\frac{\partial}{\partial y} p_s^{(r_s)}$ at points i, j, k, to obtain second-order approximation values for $\nabla\phi$ at the three vertices of the triangle $\triangle_l$. We remark that for practical implementation, we do not use the polynomial itself, but compute a series of constants $\{a_l\}_{l=1}^{m}$ which depend on the local geometry only, such that:

$$\frac{\partial}{\partial x} p_s^{(r_s)}(x_n, y_n) = \sum_{l=1}^{m} a_l \phi_l \tag{2.6}$$

where every constant a_l corresponds to one node in the stencil $\Gamma_s^{(r_s)}$ and m is the total number of nodes in $\Gamma_s^{(r_s)}$. For every vertex (x_n, y_n) of triangle $\triangle_l$, we obtain a series of such constants. And for the y directional partial derivative, we compute the corresponding constants too.

4. For every group $\{\Gamma_s^{(r_s)}, s = 1, \cdots, 5\}$, we form linear systems and solve them to get a series of linear weights $\gamma_{s,x}^{(r_s)}$ and $\gamma_{s,y}^{(r_s)}$ satisfying the equalities (2.5), for the three vertices i, j, k. Using the previous argument for combining low-order approximations to get a high-order approximation, we form the linear system for $\gamma_{s,x}^{(r_s)}$ at a vertex (ξ_n, η_n) as follows (note that we use normalized variables): take $\phi = \xi^3, \xi^2\eta, \xi\eta^2, \eta^3$ respectively, the equalities are:

$$\sum_{s=1}^{5} \gamma_{s,x}^{(r_s)} \frac{\partial}{\partial \xi} p_s^{(r_s)}(\xi_n, \eta_n) = \frac{\partial}{\partial \xi} \phi(\xi_n, \eta_n) \tag{2.7}$$

where $p_s^{(r_s)}$ is the quadratic interpolation polynomial for ϕ, using stencil $\Gamma_s^{(r_s)}$. Again, in practical implementation, we will not use $p_s^{(r_s)}$ itself, instead we use the constants computed in the last step and equation (2.6) to compute the approximation for the derivatives of ϕ. Together with the requirement

$$\sum_{s=1}^{5} \gamma_{s,x}^{(r_s)} = 1, \tag{2.8}$$

we obtain a 5×5 linear system for $\gamma_{s,x}^{(r_s)}$. For $\gamma_{s,y}^{(r_s)}$, the same argument can be applied. Note that we need to compute the reciprocal condition number c for every linear system again. If $c \geq \delta$ for the same threshold $\delta = 10^{-3}$, we will accept this group of stencils as one of the remaining candidates. Otherwise, the linear system is considered to be ill-conditioned and its corresponding group of small stencils $\{\Gamma_s^{(r_s)}, s = 1, \cdots, 5\}$ is eliminated from further consideration.

5. For each of the remaining group $\Lambda_l = \{\Gamma_s^{(r_s)}, s = 1, \cdots, 5\}$, find the minimum value γ_l of all these linear weights $\gamma_{s,x}^{(r_s)}, \gamma_{s,y}^{(r_s)}$ of the three vertices i, j, k. Then find the group of small stencils whose γ_l

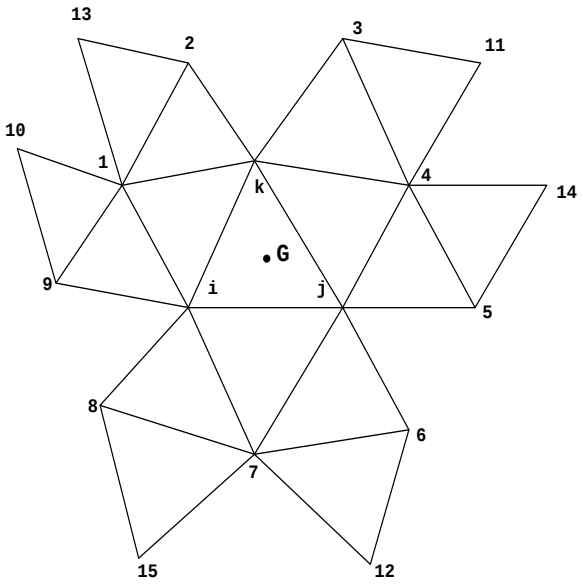

FIG. 2.3. *The nodes used for the big stencil of the fourth-order scheme.*

is the biggest among all $\{\gamma_l\}_{l=1}^{L}$, and take this group as our final 5 small stencils for triangle $\triangle_l$. Denote them by $\Gamma_s, s = 1, \cdots, 5$. For every final small stencil $\Gamma_s, s = 1, 2, \cdots, 5$, we store the index numbers of the nodes in Γ_s, the constants in the linear combinations of node values to approximate values of $\nabla\phi$ at points i, j, k, and the linear weights $\gamma_{s,x}$, $\gamma_{s,y}$ of the three points i, j, k.

6. Now we have set up the necessary constants which only depend on the mesh for all triangles. To form the final linear scheme, we compute the third-order approximations $(\nabla\phi)_0, \cdots, (\nabla\phi)_{k_l}$ for all mesh nodes l, by the linear combinations of second-order approximations, using the prestored constants and linear weights. Then we can form the scheme (2.3). ■

Remark 2.2. The strategy described above of finding second order smaller stencils to make up the third order big stencil has been reached after our extensive numerical experiments. They are found to be robust to different triangulations and equations. Of course we are not claiming that this procedure will not fail. We can only claim that it behaves nicely in all our numerical tests. The computational cost of this procedure is quite high, as many choices and comparisons are needed. Fortunately, at least for problems without adaptive mesh refinements, this procedure is done only once at the beginning, and does not need to be repeated during time marching. Because of the prestored constant coefficients for computing the the approximation to derivatives, the time evolution part of the scheme is very efficient. ■

2.2.2. Fourth-order linear schemes. To construct a fourth-order linear scheme, we need a fourth-order approximation to $\nabla\phi$. Hence a fourth degree interpolation polynomial $p^4(x, y)$ is required, which has 15 degrees of freedom. We will use part or all of the nodes shown in Figure 2.3 to construct the big stencil of the fourth-order scheme.

Comparing Figure 2.3 with Figure 2.2, we can see that, in order to get the set of candidate nodes for the big stencil of the fourth-order scheme, we have added nodes $10, 11, 12, 13, 14, 15$ (Figure 2.3) to the set for the third order scheme (Figure 2.2). Notice again that some of the added nodes may not be distinct.

The procedure to determine the big stencil of the fourth-order scheme is slightly different from that for the third-order scheme. This difference is mainly due to implementation efficiency considerations and the

more stringent stability requirement for the higher order scheme. We will not use the distances between the barycenter point and the nodes to sort our candidate nodes as in Procedure 2.1. Instead, we give an ordering to the candidate nodes. This ordering is given so that, when the nodes are chosen sequentially from it to form the big stencil S_0, the target triangle $\triangle_0$ remains central to avoid serious downwind bias which could lead to linear instability. Referring to Figure 2.3, our interpolation points for the polynomial p^4 include nodes i, j, k and the nodes taken from the sorted set: $W = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\}$. The detailed procedure to determine the big stencil S_0 for the target triangle $\triangle_0$ is given below.

Procedure 2.3: The big stencil of the fourth-order scheme.

1. Find 15 different nodes (other than the nodes i, j, k) around triangle $\triangle_0$, and give the order for them as in Figure 2.3. Naming them using their ordering number, we get the sorted node set: $W = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\}$.
2. To start with, we take $S_0 = \{i, j, k, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$. Use this stencil S_0 to form the 15×15 interpolation coefficient matrix A as in Procedure 2.1.
3. Compute the reciprocal condition number c of A. If $c \geq \delta$ for the same threshold $\delta = 10^{-3}$, we have obtained the final big stencil S_0. Otherwise, we add the next node (i.e. node 13) in W to S_0. Then we obtain the 16×15 interpolation coefficient matrix A associated with this S_0, an over-determined system which can be solved by the least square procedure. Compute the reciprocal condition number of A and check whether $c \geq \delta$ again. Repeat this if necessary until $c \geq \delta$ is satisfied. In our numerical tests, at most 16 nodes are needed in S_0 to reach our threshold value in all the cases. ■

We emphasize again that the ordering for the nodes as in Figure 2.3 is important to guarantee the chosen stencil to yield to linear stability of the scheme.

Using the big stencil, we obtain the fourth-order approximation ∇p^4 for $\nabla\phi$. The key step in building our fourth-order WENO scheme is to get a fourth-order approximation for $\nabla\phi$ based on lower order approximations. We still would like to construct several second-order approximations whose weighted average will give the same result as ∇p^4 at each angular sector of every node, i.e.,

$$\frac{\partial}{\partial x}p^4(x_l, y_l) = \sum_{s=1}^{q} \gamma_{s,x} \frac{\partial}{\partial x} p_{s,x}(x_l, y_l), \qquad \frac{\partial}{\partial y}p^4(x_l, y_l) = \sum_{s=1}^{q} \gamma_{s,y} \frac{\partial}{\partial y} p_{s,y}(x_l, y_l) \tag{2.9}$$

where $p_{s,x}, p_{s,y}$ are the quadratic interpolation polynomials, and $\gamma_{s,x}, \gamma_{s,y}$ are the linear weights.

Remark 2.3. The reason that we still use second-order approximations, not third-order ones, as the lower order approximations is that each second-order approximation needs fewer nodes in the big stencil, thus it is easier to make the small stencils to be sufficiently diversified, which is important for the WENO technique to function well near discontinuous derivatives in the solutions. We have encountered difficulties in obtaining non-oscillatory results converging to the correct viscosity solutions for some of the more demanding test cases when third order building blocks are used instead of the second order ones. ■

We now determine the value of q, which is the number of small stencils, in the equalities (2.9). Because p^4 has nine more degrees of freedom than the quadratic polynomials $p_{s,x}, p_{s,y}$, i.e., $x^3, x^2y, xy^2, y^3, x^4, x^3y, x^2y^2$, xy^3, y^4, according to our previous argument, we would need $q \geq 10$. We take $q = 10$ in our fourth-order scheme. The procedure to find all the small stencils is described below.

Procedure 2.4: The fourth-order linear scheme.

For every triangle $\triangle_l, l = 1, \cdots, N$, do steps $1 \sim 3$:

1. Follow Procedure 2.3 to obtain the big stencil S_l for $\triangle_l$.

2. For each of the six approximations (ϕ_x, ϕ_y at vertices i, j, k of $\triangle_l$), find 10 small stencils respectively. Let $\{\Gamma_{s,x}, s = 1, \cdots, 10\}$ denote the small stencils to approximate ϕ_x at vertex i. To find them, we first determine 10 preliminary small stencils $\{\Gamma_s^0, s = 1, \cdots, 10\}$ as follows (see Figure 2.3):
   (1) Include nodes $\{i, j, k, 1, 4, 7\}$ in Γ_1^0.
   (2) Include nodes $\{i, j, k, 1, 2, 9\}$ in Γ_2^0.
   (3) Include nodes $\{i, j, k, 4, 3, 5\}$ in Γ_3^0.
   (4) Include nodes $\{i, j, k, 7, 8, 6\}$ in Γ_4^0.
   (5) Include nodes $\{i, j, k\}$ in all $\Gamma_s^0, s = 5, 6, \cdots, 10$.
   (6) Take points $A_s, s = 1, \cdots, 10$ as the center of $\{\Gamma_s^0\}_{s=1}^{10}$, where A_1 = point G, A_2 = node 1, A_3 = node 4, A_4 = node 7, A_5 = node 10, A_6 = node 13, A_7 = node 11, A_8 = node 14, A_9 = node 12, and A_{10} = node 15.
   (7) Obtain 6 nodes for each of $\{\Gamma_s^0\}_{s=1}^{10}$. There are already 6 points in $\{\Gamma_s^0\}_{s=1}^{4}$. For $\{\Gamma_s^0\}_{s=5}^{10}$, we add 3 nodes (other than i, j, k) from S_l to each of them, which have the shortest distances from A_s. Then use Γ_s^0 to get the interpolation coefficient matrix, and judge the reciprocal condition number c of the matrix. If c reaches our threshold value $\delta = 10^{-3}$, then Γ_s^0 is found. Otherwise add one different node from S_l to Γ_s^0, which has the shortest distance from A_s. Continue this until $c \geq \delta$ is satisfied.
   (8) All of the 6 approximations (to ϕ_x, ϕ_y at vertices i, j, k of $\triangle_l$) have the common 10 preliminary small stencils $\{\Gamma_s^0, s = 1, \cdots, 10\}$.

   Now we come to determine the $\{\Gamma_{s,x}, s = 1, \cdots, 10\}$. Our goal is that the coefficient matrix $\widehat{A}$ of the 10×10 linear system, which is obtained from $\{\Gamma_{s,x}\}_{s=1}^{10}$ and used to compute the linear weights, has a good condition number. For $s = 1, 2, \cdots, 10$, we perform the following steps:
   (1) Taking the original small stencil Γ_s^0 as the interpolation stencil, compute the constants $\{a_l\}_{l=1}^{m}$, which depend on the mesh only and are the coefficients in the linear combination of function values at the nodes in Γ_s^0 to get the second-order approximation to ϕ_x at vertex i.
   (2) Form the s-th column of the matrix $\widehat{A}$. Let $\widehat{a}_{rt}$ be the elements of the matrix $\widehat{A}$. Take the 9 functions $\{\phi^{(r)}\}_{r=2}^{10} = \xi^3, \xi^2\eta, \xi\eta^2, \eta^3, \xi^4, \xi^3\eta, \xi^2\eta^2, \xi\eta^3, \eta^4$ respectively, and let $p_{s,x}^{(r)}$ be the second-order interpolation polynomial for $\phi^{(r)}, r = 2, \cdots, 10$, then $\widehat{a}_{1s} = 1$, and

$$a_{rs} = \frac{\partial}{\partial \xi} p_{s,x}^{(r)}(\xi_i, \eta_i), \qquad r = 2, \cdots, 10 \tag{2.10}$$

   where (ξ_i, η_i) is the coordinate of vertex i (again note that we use normalized variables). We use the constants computed at last step to implement this, and will not need to use the polynomials themselves.
   (3) Now $\widehat{A}$ has s columns and is a $10 \times s$ matrix. Compute its reciprocal condition number c. If $c \geq \delta$, take the current small stencil as our s-th small stencil $\Gamma_{s,x}$. Otherwise, change one node which is in the current small stencil and farthest from the center point A_s, to another node from S_l which is not in the current small stencil but is nearest to A_s. Now we get a new small stencil. This part can be repeated if $c \geq \delta$ is still not satisfied. Then the current small stencil is our s-th small stencil.

   For the small stencils to approximate the y-directional derivative at vertex i and x,y-directional derivatives at other 2 vertices j, k, we use a similar procedure.
3. Now we have obtained the coefficient matrix $\widehat{A}$ with a good condition number. Along with the right

hand vector $\vec{b}$, whose components are $b_1 = 1$ and

$$b_r = \frac{\partial}{\partial \xi}\phi^{(r)}(\xi_i, \eta_i), \qquad r = 2, \cdots, 10, \tag{2.11}$$

we obtain the 10×10 linear system

$$\widehat{A}\vec{\gamma} = \vec{b}. \tag{2.12}$$

We solve it to get the linear weights $\vec{\gamma} = (\gamma_{1,x}, \cdots, \gamma_{10,x})^T$. We use similar method to get $\{\gamma_{s,y}\}_{s=1}^{10}$. Then for each of small stencils, we store the index numbers of the nodes in it, the constants in the linear combinations of node values to approximate values of the derivatives of ϕ, and the linear weights.

4. Now we have set up necessary constants which only depend on the mesh for all triangles. To form the final linear scheme, we compute the fourth-order approximation $(\nabla\phi)_0, \cdots, (\nabla\phi)_{k_l}$ for all the mesh nodes l, by the linear combinations of second-order approximations, using the prestored constants and linear weights. We then form the scheme (2.3). ■

Remark 2.4. We notice that in the fourth-order scheme, although the big stencil is the same to approximate $\frac{\partial}{\partial x}\phi$ and $\frac{\partial}{\partial y}\phi$ at all three vertices i, j, k of the target cell $\triangle_0$, the small stencils can be different for the x-direction and y-direction derivatives at one vertex, and also different at the three vertices i, j, k in the same cell $\triangle_0$. This strategy is different from that for the third-order scheme described before. The reason for this difference is that we need more small stencils than in the third-order case and it is difficult to find a group of common small stencils for all the six cases (three vertices, 2 derivatives each). ■

Remark 2.5. Again, the procedure described above of finding second order smaller stencils to make up the fourth order big stencil has been reached after extensive numerical experiments. They are found to be robust to different triangulations and equations. Of course we are not claiming that this procedure will not fail. We can only claim that it behaves nicely in all our numerical tests. The computational cost of this procedure is again quite high, as many choices and comparisons are needed. Fortunately, at least for problems without adaptive mesh refinements, this procedure is done only once at the beginning, and does not need to be repeated during time marching. Because of the prestored constant coefficients for computing the the approximation to derivatives, the time evolution part of the scheme is again very efficient. ■

2.3. WENO schemes. In this section, we construct the WENO schemes based on non-linear weights. The resulting schemes will be suitable to compute the H-J equations whose solutions are not smooth.

2.3.1. WENO approximation. We only discuss the case of WENO approximation for the x-directional derivative at vertex i of the target cell $\triangle_l$. Other cases are similar. In order to compute the non-linear weights, we need to compute the smoothness indicators first.

For a polynomial $p(x, y)$ defined on the target cell $\triangle_0$ with degree up to k, we take the smoothness indicator β as:

$$\beta = \sum_{2 \le |\alpha| \le k} \int_{\triangle_0} |\triangle_0|^{|\alpha|-1} \left(D^\alpha p(x, y)\right)^2 dxdy \tag{2.13}$$

where α is a multi-index and D is the derivative operator. The smoothness indicator measures how smooth the function p is on the triangle $\triangle_0$: the smaller the smoothness indicator, the smoother the function p is on $\triangle_0$. The scaling factor in front of the derivatives renders the smoothness indicator self-similar and invariant under uniform scaling of the mesh in all directions.

Remark 2.6. The definition of the smoothness indicator in equation (2.13) is different from that used in the WENO schemes for conservation laws [8]. In equation (2.13), the range of summation is $2 \leq |\alpha| \leq k$, but in [8] for conservation laws, it is $1 \leq |\alpha| \leq k$. An intuitive reason is that H-J equations are in some sense the conservation laws "integrated once", hence smooth indicators for the former should involve derivatives one order higher than those for the latter. A similar strategy is also used in ENO schemes for H-J equations in [13]. We have found through numerical experiments, that if we still take $1 \leq |\alpha| \leq k$ to compute the smoothness indicators, we cannot obtain the correct viscosity solutions for some non-convex Hamilton-Jacobi equations. ■

Now we define the non-linear weights as:

$$\omega_j = \frac{\widetilde{\omega}_j}{\sum_m \widetilde{\omega}_m}, \qquad \widetilde{\omega}_j = \frac{\gamma_j}{(\varepsilon + \beta_j)^2} \tag{2.14}$$

where γ_j is the jth linear weight (e.g. the $\gamma_{s,x}$ in our linear schemes), β_j is the smoothness indicator for the jth interpolation polynomial $p_j(x, y)$ (e.g. the p_s in equation (2.5) for the third-order case and the $p_{s,x}$ in equation (2.9) for the fourth-order case) associated with the jth small stencil, and ε is a small positive number to avoid the denominator becoming 0. We take $\varepsilon = 10^{-6}$ for all the computations in this paper. The final WENO approximation for the x-directional derivative at vertex i of target cell $\triangle_l$ is given by

$$(\phi_x)_i = \sum_{j=1}^{q} \omega_j \frac{\partial}{\partial x} p_j(x_i, y_i) \tag{2.15}$$

where (x_i, y_i) are the coordinates of vertex i, $q = 5$ for the third-order schemes and $q = 10$ for the fourth-order schemes.

In our WENO schemes, the linear weights $\{\gamma_j\}_{j=1}^q$ depend on the local geometry of the mesh and can be negative. If $min(\gamma_1, \cdots, \gamma_q) < 0$, we adopt the splitting technique of treating negative weights in WENO schemes developed by Shi, Hu and Shu [15]: first we split the linear weights into two groups:

$$\widetilde{\gamma}_j^+ = \frac{1}{2}(\gamma_j + 3|\gamma_j|), \qquad \widetilde{\gamma}_j^- = \widetilde{\gamma}_j^+ - \gamma_j, \qquad j = 1, \cdots, q \tag{2.16}$$

then we scale them by

$$\sigma^{\pm} = \sum_{l=1}^{q} \widetilde{\gamma}_l^{\pm}; \qquad \gamma_j^{\pm} = \widetilde{\gamma}_j^{\pm} / \sigma^{\pm}, \qquad j = 1, \cdots, q. \tag{2.17}$$

Now we compute the nonlinear weights (2.14) for the positive and negative groups $\gamma_j^{\pm}$ separately, denoted by $\omega_j^{\pm}$, based on the same smoothness indicator β_j. Then we compute the WENO approximations $(\phi_x)_i^{\pm}$ separately by (2.15), using $\omega_j^{\pm}$, and form the final WENO approximation by

$$(\phi_x)_i = \sigma^+ (\phi_x)_i^+ - \sigma^- (\phi_x)_i^-. \tag{2.18}$$

The key idea of this decomposition is to make sure that every stencil has a significant representation in both the positive and the negative weight groups. Within each group, the WENO idea of redistributing the weights subject to a fixed sum according to the smoothness of the approximation is still followed as before. See [15] for more details.

Again, we remark that the smoothness indicator (2.13) is a quadratic function of function values on nodes of the small stencil, so in a practical implementation, to compute the smoothness indicator β_j for the j-th small stencil by equation (2.13), we do not need to use the interpolation polynomial itself, instead we

use a series of constants $\{a_{rt}, r = 1, \cdots, t; t = 1, \cdots, m\}$, which can be precomputed and they depend on the mesh only, such that

$$\beta_j = \sum_{t=1}^{m} \phi_t (\sum_{r=1}^{t} a_{rt} \phi_r), \tag{2.19}$$

where m is the total number of nodes in the j-th small stencil. These constants for all smoothness indicators should be precomputed and stored once the mesh is generated.

2.3.2. Algorithm flowchart. We summarize the algorithm for the third-order and the fourth-order WENO schemes as follows:

Procedure 2.5: The third- and fourth-order WENO schemes.

1. Generate a triangular mesh.
2. Compute and store all constants which only depend on the mesh and the accuracy order of the scheme. These constants include the node index numbers of each small stencil, the coefficients in the linear combinations of function values on nodes of small stencil to approximate the derivative values and the linear weights, following the Procedure 2.2 for the third-order case and the Procedure 2.4 for the fourth-order case, and the constants for computing smoothness indicators in equation (2.19).
3. Using the prestored constants, for each angular sector of every node i, compute the low-order approximations for $\nabla\phi$ and nonlinear weights, then compute the third or fourth-order WENO approximations (2.15) or (2.18). Form the scheme (2.3). Use high-order TVD Runge-Kutta time stepping [16] to evolve in time.

3. Numerical examples. In this section, we apply the WENO schemes developed in the previous section to a set of one and two dimensional problems. The CFL number is taken as 0.5 in all the cases, except for the accuracy test where it is taken to be smaller if necessary to guarantee that spatial errors dominate. For the temporal discretization, we use the third-order TVD Runge-Kutta scheme of Shu and Osher in [16].

We have not described the details of the one dimensional algorithm in the previous section. The algorithm in 1D is just the same 2D algorithm with only two "angular sectors". The WENO interpolation follows along the lines of [8] and [3]. In all the one-dimensional numerical examples, we use non-uniform meshes. These non-uniform meshes are obtained by randomly shifting the cell boundaries in a uniform mesh in the range $[-0.1h, 0.1h]$ where h is the uniform mesh size. When we perform the accuracy test, the refinement of the meshes is achieved by cutting each cell into two smaller equal sized ones.

Example 3.1. One-dimension linear equation:

$$\begin{cases} \phi_t + \phi_x = 0, & 0 \le x < 2\pi, \\ \phi(x, 0) = \sin(x) \end{cases} \tag{3.1}$$

with periodic boundary conditions. We remark that even though this equation is also a conservation law, the method used here is different from the WENO schemes for conservation laws in [8].

We use third, fifth and seventh order linear and WENO schemes to compute the problem to $t = 2.0$. The errors and the numerical orders of accuracy are listed in Table 3.1. We can see that the correct orders of

TABLE 3.1
Accuracy for 1D linear equation, Linear and WENO Schemes, t=2

Linear	3rd order		5th order		7th order	
N	L^∞ error	order	L^∞ error	order	L^∞ error	order
10	4.38E-02	—	4.94E-03	—	1.23E-03	—
20	5.85E-03	2.91	1.68E-04	4.88	1.06E-05	6.86
40	7.44E-04	2.97	5.56E-06	4.92	8.63E-08	6.94
80	9.41E-05	2.98	1.79E-07	4.95	6.91E-10	6.96
160	1.18E-05	2.99	5.67E-09	4.98	5.46E-12	6.98

WENO	3rd order		5th order		7th order	
N	L^∞ error	order	L^∞ error	order	L^∞ error	order
10	1.55E-01	—	2.00E-02	—	3.07E-03	—
20	4.64E-02	1.74	8.64E-04	4.53	3.45E-05	6.48
40	1.22E-02	1.93	2.76E-05	4.97	4.24E-07	6.35
80	1.66E-03	2.88	8.75E-07	4.98	4.04E-09	6.71
160	5.38E-05	4.95	2.33E-08	5.23	1.04E-11	8.61

accuracy are obtained by both the linear and WENO schemes. In all the accuracy tests, we only list results for L^∞ errors. Those for L^1 or L^2 errors, which follow the same patterns, are omitted to save space.

Example 3.2. One-dimensional linear equation:

$$\begin{cases} \phi_t + \phi_x = 0, & -1 \le x < 1, \\ \phi(x,0) = g(x-0.5) \end{cases} \tag{3.2}$$

with periodic boundary conditions. Where

$$g(x) = -(\frac{\sqrt{3}}{2} + \frac{9}{2} + \frac{2\pi}{3})(x+1) + \begin{cases} 2\cos(\frac{3\pi x^2}{2}) - \sqrt{3}, & -1 \le x < -\frac{1}{3}; \\ \frac{3}{2} + 3\cos(2\pi x), & -\frac{1}{3} \le x < 0; \\ \frac{15}{2} - 3\cos(2\pi x), & 0 \le x < \frac{1}{3}; \\ \frac{28+4\pi+\cos(3\pi x)}{3} + 6\pi x(x-1), & \frac{1}{3} \le x < 1. \end{cases} \tag{3.3}$$

This example comes from [7]. We use third, fifth and seventh order WENO schemes with 101 non-uniformly spaced points and show the results at t=2 and 8 in Figure 3.1. We can clearly observe better resolution with increased order of accuracy.

Example 3.3. One-dimensional Burgers equation:

$$\begin{cases} \phi_t + \frac{(\phi_x+1)^2}{2} = 0, & -1 \le x < 1, \\ \phi(x,0) = -\cos(\pi x) \end{cases} \tag{3.4}$$

with periodic boundary conditions.

The local Lax-Friedrichs flux (2.4) is used. At $t = 0.5/\pi^2$, the solution is still smooth. We list the errors and the numerical orders of accuracy in Table 3.2, using linear and WENO schemes. We see that the correct

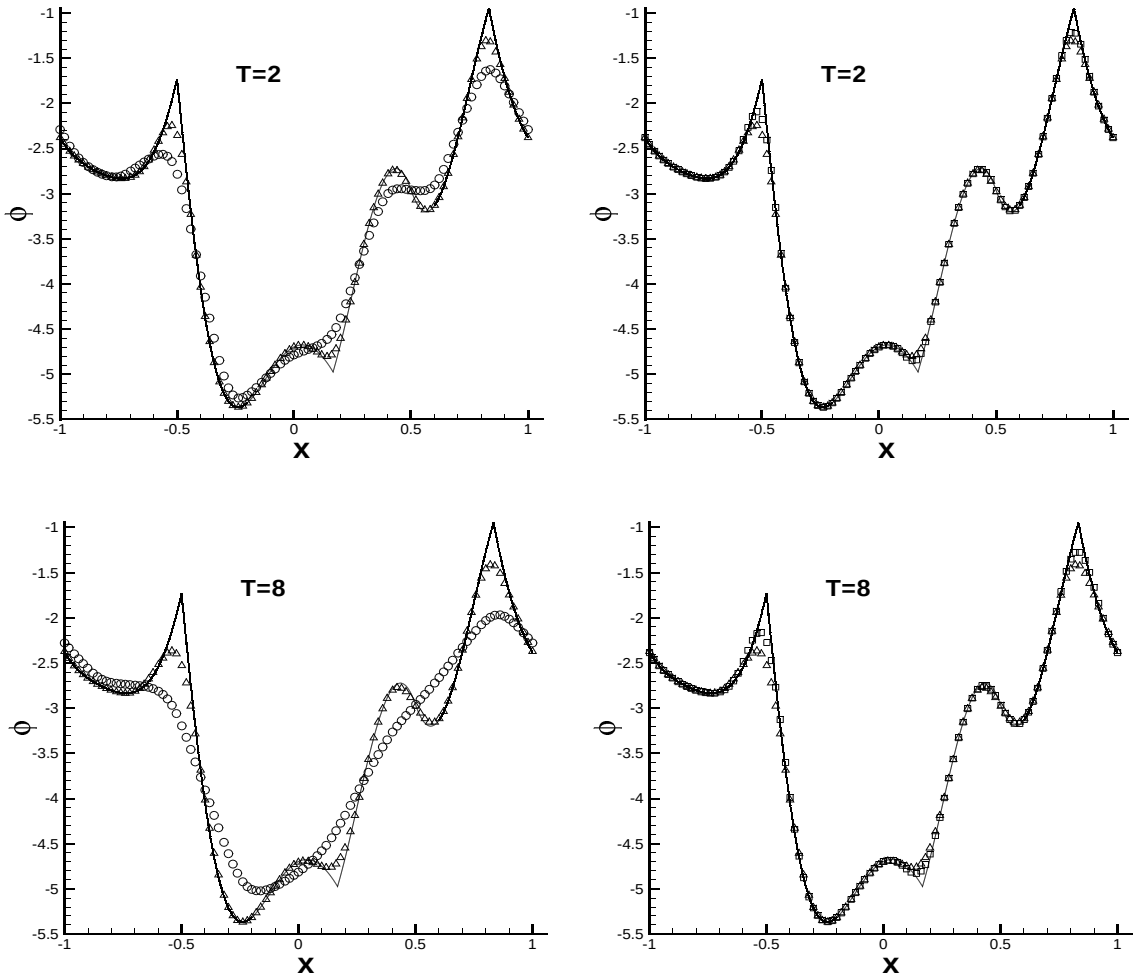

FIG. 3.1. *One-dimensional linear equation. Non-uniform mesh with 101 points. Solid line is the exact solution. Left: a comparison of third order 'o' and fifth order 'Δ'; right: a comparison of fifth order 'Δ' and seventh order '□'. Top: results at $t = 2$; bottom: results at $t = 8$.*

orders of accuracy are obtained. At $t = 3.5/\pi^2$, the solution has developed a discontinuous derivative. In Figure 3.2, we show the sharp corner-like numerical solution with 41 points using third, fifth and seventh order WENO schemes. From now on, the solid line is the exact solution, and the circles are numerical solutions.

Example 3.4. One-dimensional equation with a non-convex flux:

$$\begin{cases} \phi_t - \cos(\phi_x + 1) = 0, & -1 \le x < 1, \\ \phi(x, 0) = -\cos(\pi x) \end{cases} \tag{3.5}$$

with periodic boundary conditions.

The local Lax-Friedrichs flux (2.4) is used. At $t = 1.5/\pi^2$, the solution has developed a corner-like discontinuity in the derivative. The numerical result with 41 points is shown in Figure 3.3.

TABLE 3.2
Accuracy for 1D Burgers equation, Linear and WENO Schemes, $t = 0.5/\pi^2$.

Linear	3rd order		5th order		7th order	
N	L^∞ error	order	L^∞ error	order	L^∞ error	order
10	8.12E-03	—	3.89E-03	—	1.75E-03	—
20	1.46E-03	2.47	4.29E-04	3.18	1.14E-04	3.95
40	2.52E-04	2.54	2.33E-05	4.20	4.78E-06	4.57
80	3.83E-05	2.72	8.94E-07	4.70	8.54E-08	5.81
160	5.23E-06	2.87	2.94E-08	4.93	8.57E-10	6.64
320	6.82E-07	2.94	9.39E-10	4.97	7.33E-12	6.87

WENO	3rd order		5th order		7th order	
N	L^∞ error	order	L^∞ error	order	L^∞ error	order
10	5.29E-02	—	5.21E-03	—	2.92E-03	—
20	1.93E-02	1.45	3.78E-04	3.78	3.02E-04	3.27
40	4.76E-03	2.02	3.05E-05	3.63	5.58E-06	5.76
80	5.97E-04	3.00	1.26E-06	4.60	5.68E-08	6.62
160	3.03E-05	4.30	4.19E-08	4.91	5.66E-10	6.65
320	1.11E-06	4.77	9.18E-10	5.51	6.28E-12	6.50

Example 3.5. The one-dimensional Riemann problem with a non-convex flux:

$$\begin{cases} \phi_t + \frac{1}{4}(\phi_x^2 - 1)(\phi_x^2 - 4) = 0, & -1 < x < 1, \\ \phi(x, 0) = -2|x|. \end{cases} \tag{3.6}$$

We remark that this is a demanding test case. Many schemes have poor resolutions or could even converge to a non-viscosity solution for this case. The global Lax-Friedrichs flux (2.4) is used. Local Lax-Friedrichs flux is less satisfactory for this case. Numerical results at $t = 1$ with 81 non-uniform grid points are shown in Figure 3.4. Third, fifth and seventh order WENO schemes are used.

Example 3.6. Two dimensional linear equation:

$$\begin{cases} \phi_t + \phi_x + \phi_y = 0, & -2 \le x < 2, -2 \le y < 2, \\ \phi(x, y, 0) = \sin(\frac{\pi}{2}(x + y)). \end{cases} \tag{3.7}$$

with periodic boundary conditions.

We first use uniform triangular meshes, shown in Figure 3.5 for the coarsest case $h = \frac{2}{5}$ where h is the length of the right angled side, to test the accuracy for the third and fourth order linear schemes and WENO schemes. We then use non-uniform meshes, shown in Figure 3.6 for the coarsest case $N = 105$ where N is the total number of nodes in the mesh. The refinement of the non-uniform meshes is done in a uniform way, namely by cutting each triangle into four smaller similar ones. The accuracy results are shown only for the non-uniform meshes, to save space, in Table 3.3. The expected orders of accuracy are obtained.

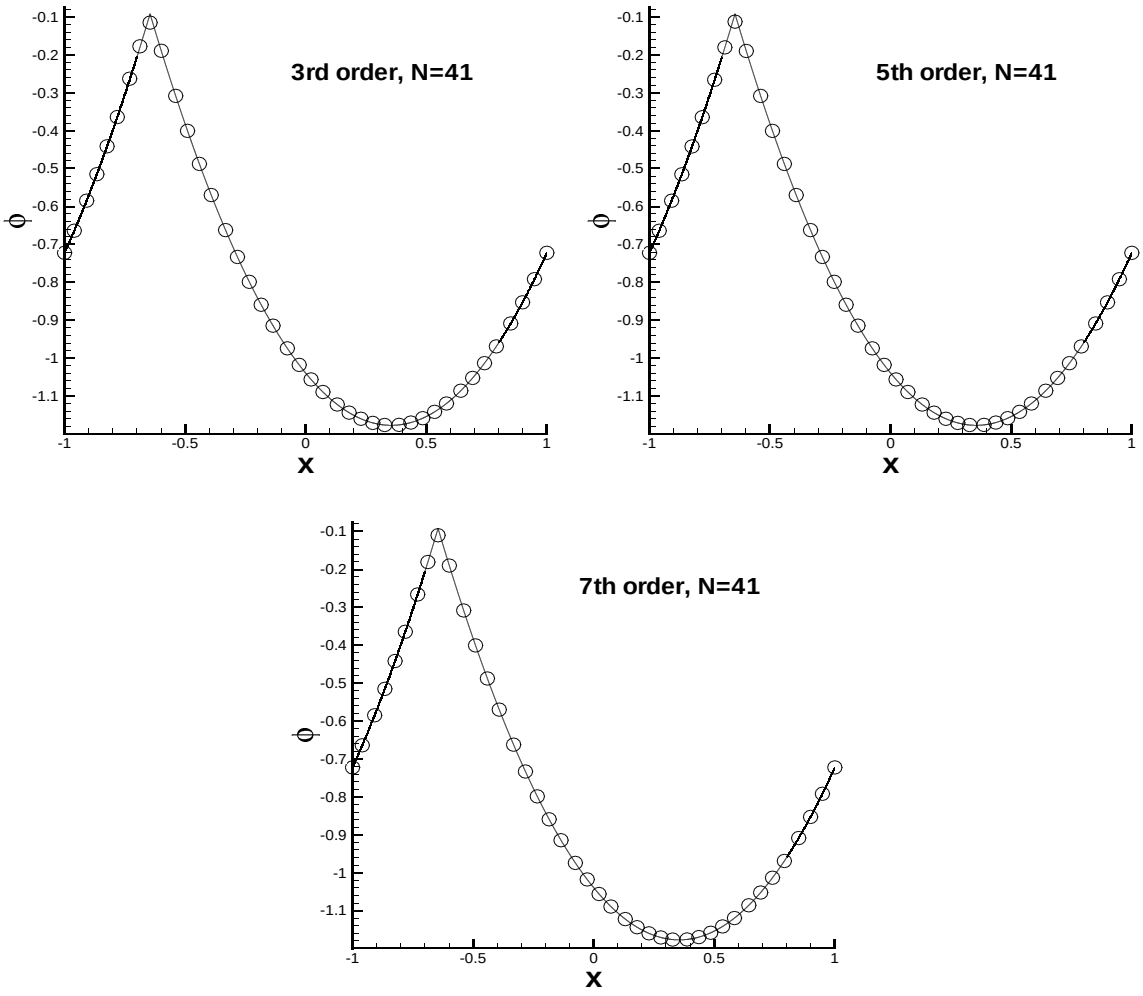

FIG. 3.2. *One-dimensional Burgers equation, local Lax-Friedrichs flux, $t = 3.5/\pi^2$. Solid line: the exact solution; circles: numerical solutions. Non-uniform mesh with 41 points. Top left: third order WENO scheme; top right: fifth order WENO scheme; bottom: seventh order WENO scheme.*

TABLE 3.3
Accuracy for 2D Linear equation, Non-uniform Meshes, Linear and WENO Schemes, $t = 2$.

	3rd order				4th order			
	Linear		WENO		Linear		WENO	
N	L^∞ error	order	L^∞ error	order	L^∞ error	order	L^∞ error	order
105	3.27E-01	—	7.00E-01	—	1.10E-01	—	6.94E-01	—
385	5.59E-02	2.55	2.42E-01	1.53	6.99E-03	3.98	2.35E-01	1.56
1473	8.04E-03	2.80	6.44E-02	1.91	4.20E-04	4.06	4.72E-02	2.32
5761	1.07E-03	2.91	1.16E-02	2.48	2.54E-05	4.05	3.20E-03	3.88
22785	1.39E-04	2.94	5.03E-04	4.52	1.49E-06	4.09	4.43E-05	6.18

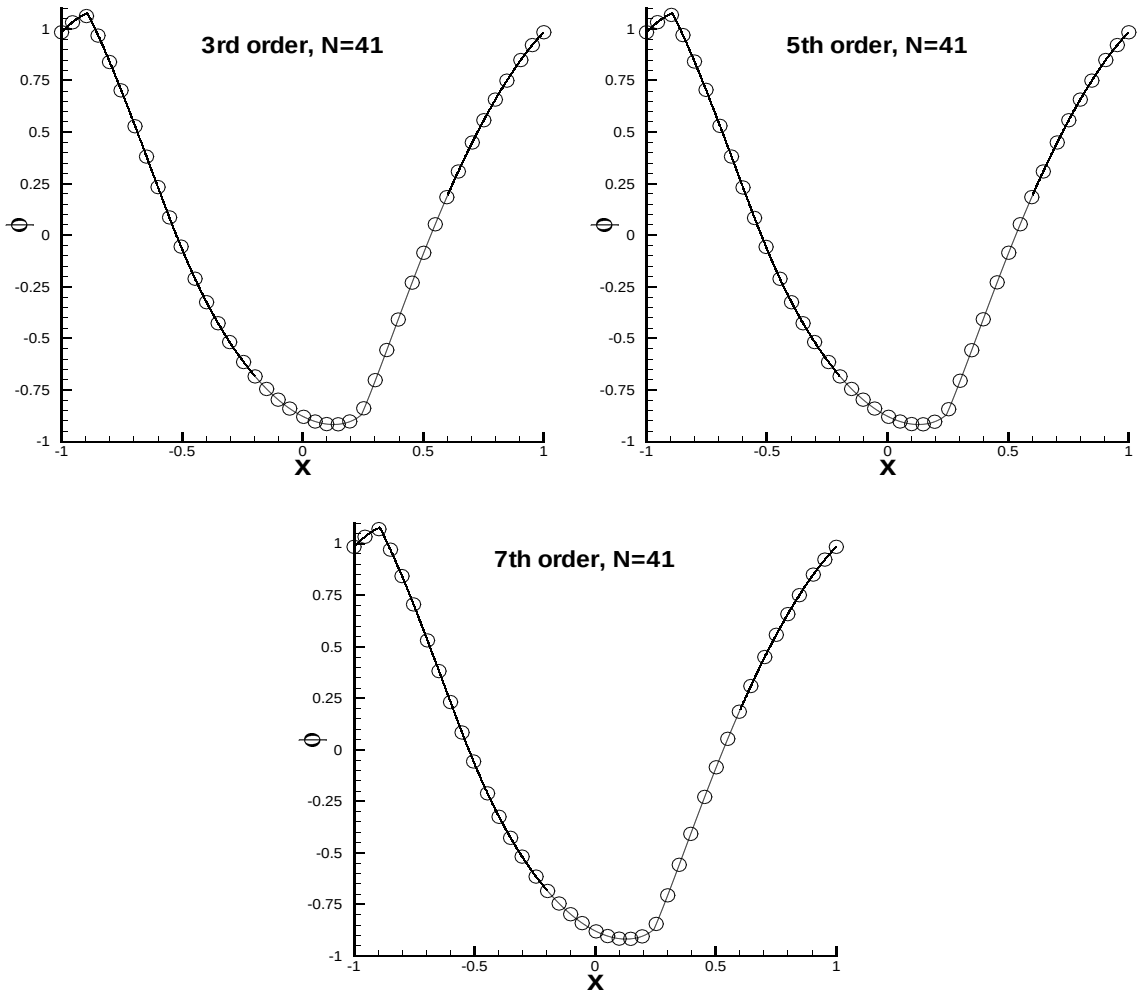

FIG. 3.3. *One dimension, non-convex, local Lax-Friedrichs flux,* $H(u) = -\cos(u+1), t = 1.5/\pi^2$. *Solid line: the exact solution; circles: numerical solutions. Non-uniform mesh with 41 points. Top left: third order WENO scheme; top right: fifth order WENO scheme; bottom: seventh order WENO scheme.*

Example 3.7. Two dimensional Burgers equation:

$$\begin{cases} \phi_t + \dfrac{(\phi_x^2 + \phi_y^2 + 1)^2}{2} = 0, & -2 \le x < 2, -2 \le y < 2, \\ \phi(x, y, 0) = -\cos(\frac{\pi(x+y)}{2}). \end{cases} \tag{3.8}$$

with periodic boundary conditions.

At $t = 0.5/\pi^2$, the solution is still smooth. We use both the uniform meshes (Figure 3.5) and the non-uniform meshes (Figure 3.6) to test the accuracy, but errors and orders of accuracy for the third and fourth order linear and WENO schemes only for the non-uniform meshes are listed in Table 3.4, to save space. We can see that the correct orders of accuracy are obtained. At $t = 1.5/\pi^2$, the solution has developed discontinuous derivatives. We use the third and fourth order linear and WENO schemes to compute, with uniform mesh of $h = \frac{1}{10}$. The results are shown in Figures 3.7 and 3.8. We can see that for this example which is not very demanding, the linear schemes which do not use WENO nonlinear weights can also obtain non-oscillatory results. It seems that the nonlinear WENO strategy is needed only for the more demanding cases such as those for some non-convex fluxes.

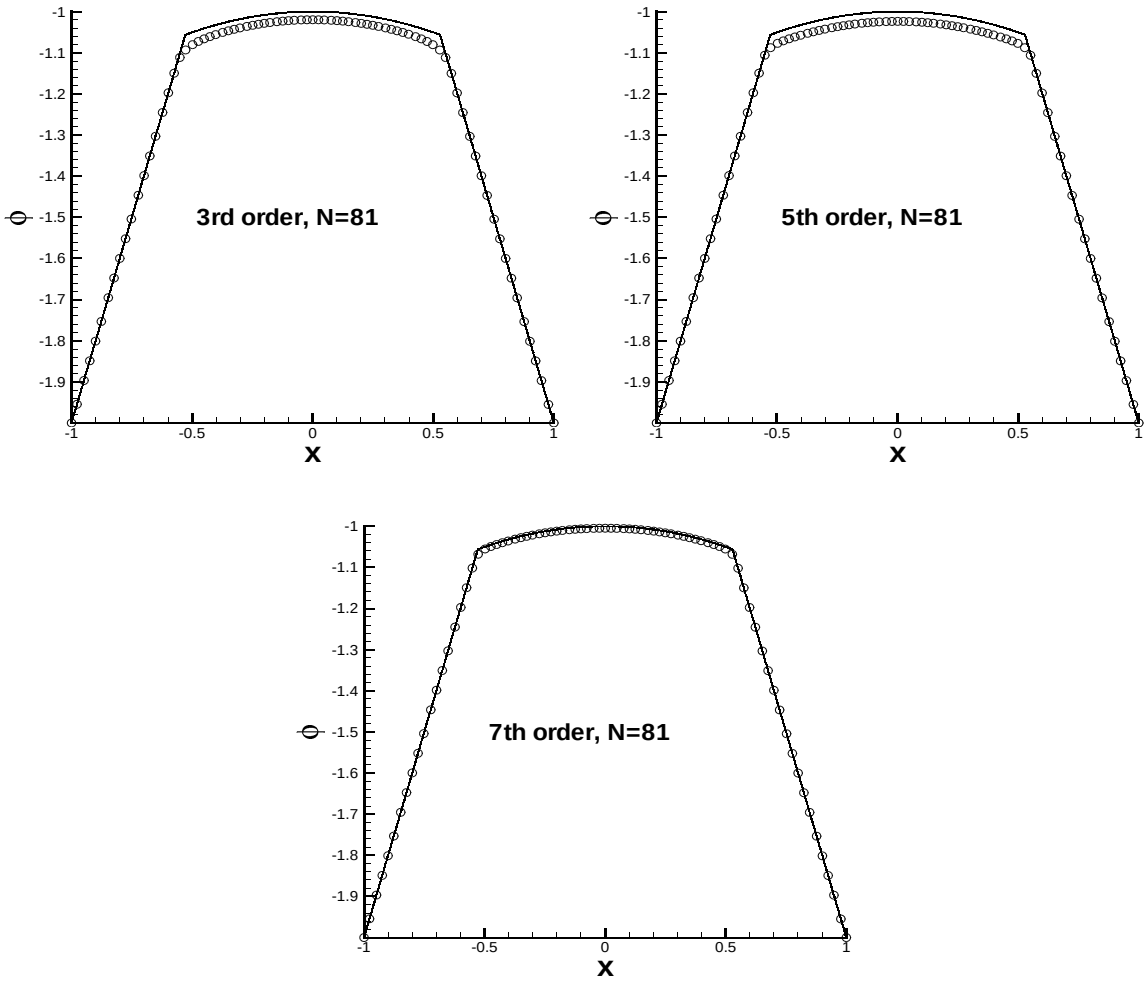

FIG. 3.4. *One-dimensional Riemann problem, global Lax-Friedrichs flux,* $H(u) = \frac{1}{4}(u^2-1)(u^2-4)$, $t = 1$. *Solid line: the exact solution; circles: numerical solutions. Non-uniform mesh with 81 points. Top left: third order WENO scheme; top right: fifth order WENO scheme; bottom: seventh order WENO scheme.*

TABLE 3.4
Accuracy for 2D Burgers equation, Non-uniform Meshes, Linear and WENO Schemes, $t = 0.5/\pi^2$.

	3rd order				4th order			
	Linear		WENO		Linear		WENO	
N	L^∞ error	order	L^∞ error	order	L^∞ error	order	L^∞ error	order
105	6.10E-02	—	1.68E-01	—	2.91E-02	—	1.71E-01	—
385	1.68E-02	1.86	5.14E-02	1.71	6.10E-03	2.26	5.61E-02	1.61
1473	3.54E-03	2.25	1.51E-02	1.77	7.45E-04	3.03	1.63E-02	1.78
5761	5.63E-04	2.65	2.76E-03	2.45	5.66E-05	3.72	1.88E-03	3.12
22785	7.70E-05	2.87	1.63E-04	4.08	2.65E-06	4.42	8.64E-05	4.45

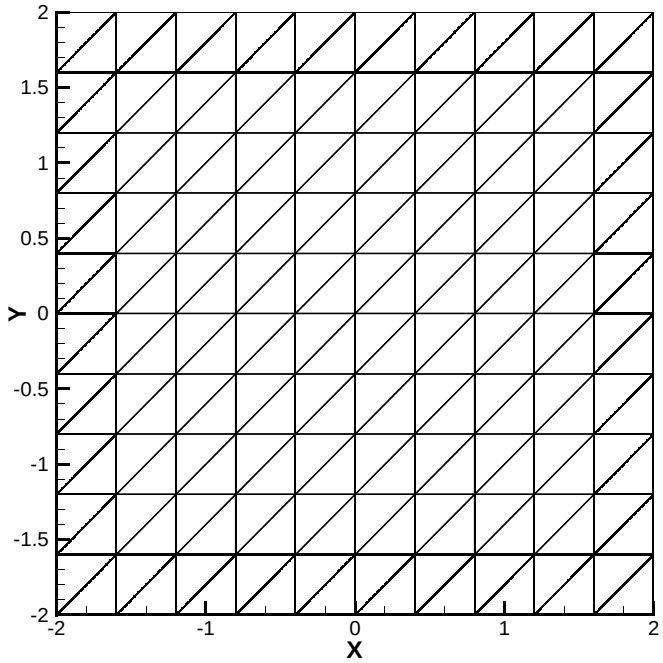

FIG. 3.5. *Uniform mesh with* $h = \frac{2}{5}$.

Example 3.8. Two-dimensional equation with a non-convex flux:

$$\begin{cases} \phi_t - \cos(\phi_x + \phi_y + 1) = 0, & -2 \le x < 2, -2 \le y < 2, \\ \phi(x, y, 0) = -\cos(\frac{\pi(x+y)}{2}). \end{cases} \tag{3.9}$$

with periodic boundary conditions.

For this example we use a non-uniform mesh (Figure 3.6) with $N = 1473$ nodes. At $t = 1.5/\pi^2$, the solution develops a discontinuous derivative, and we show the results using the third and fourth order linear and WENO schemes in Figures 3.9 and 3.10. Similar to the Burgers equation case, linear schemes which do not use WENO nonlinear weights can produce non-oscillatory results as well.

Example 3.9. The two-dimensional methods are applied to the one-dimensional non-convex Riemann problem in Example 3.5. This example is more demanding than the previous two examples, and we will see that the linear scheme will not work. We solve the problem in Example 3.5 in the domain $[-1, 1] \times [-0.2, 0.2]$ with the triangulation shown in Figure 3.11. The periodic boundary condition is applied in the y-direction. We plot the solutions along the central cut line $y = 0$.

First we use the third-order linear scheme to compute this problem, and refine the mesh to test convergence. The results are plotted in Figure 3.12, with $h = \frac{1}{20}, \frac{1}{40}, \frac{1}{80}$. We can see that the solutions of the linear scheme with one mesh refinement does not converge to the correct viscosity solution.

Next we use the third-order and fourth-order WENO schemes. From Figure 3.13, we can see that the solutions of WENO schemes converge to the correct viscosity solution when the mesh is refined. And obviously, the fourth-order scheme has a better resolution than the third-order scheme.

At last we use the first-order monotone scheme with the monotone Lax-Friedrichs Hamiltonian (2.2) to compute the problem. The results are shown in Figure 3.14. It converges to the correct viscosity solution,

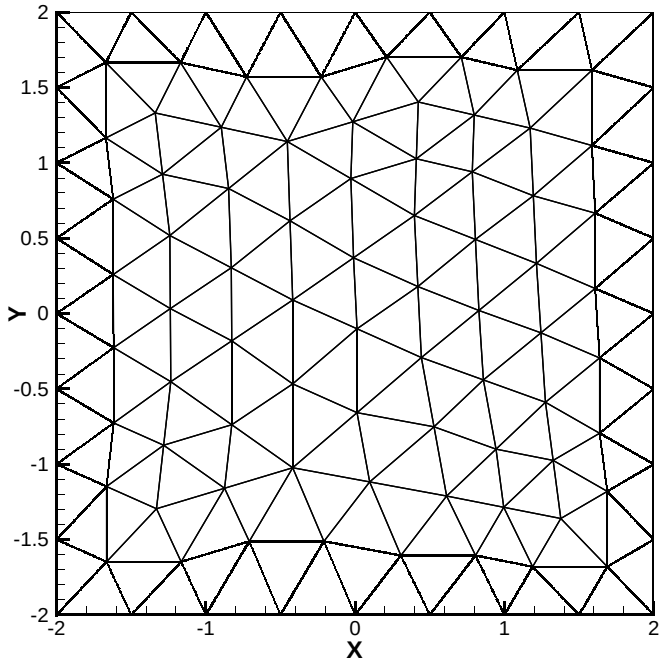

FIG. 3.6. *Non-uniform mesh for the coarsest case with $N = 105$ nodes.*

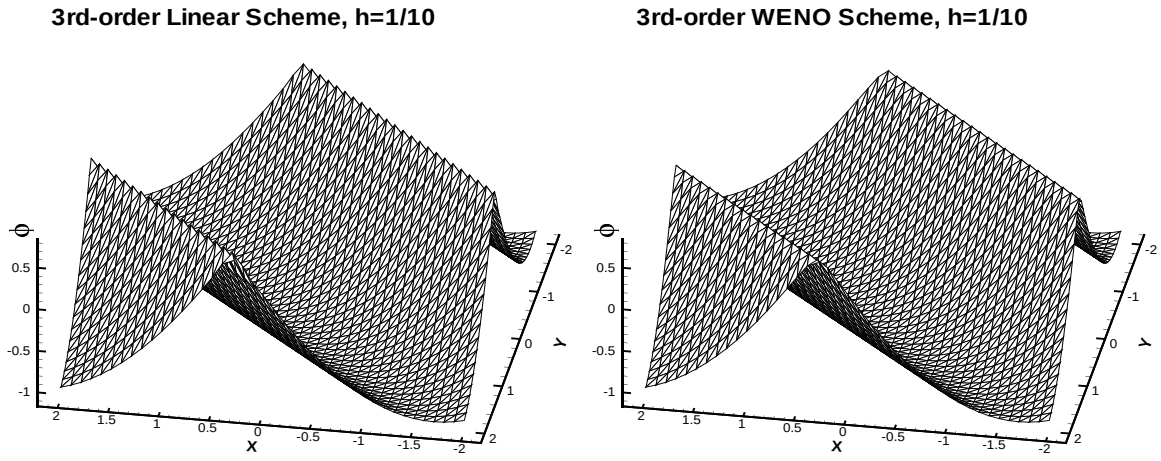

FIG. 3.7. *Two-dimensional Burgers equation, $t = 1.5/\pi^2$. Uniform mesh with $h = \frac{1}{10}$. Left: third order linear scheme; right: third order WENO scheme.*

as expected. But comparing with the results of the third and fourth order WENO schemes, the first order monotone scheme needs a much more refined mesh to achieve the same resolution.

Example 3.10. Two dimensional Riemann problem:

$$\begin{cases} \phi_t + \sin(\phi_x + \phi_y) = 0, & -1 < x < 1, -1 < y < 1, \\ \phi(x, y, 0) = \pi(|y| - |x|). \end{cases} \tag{3.10}$$

For this example, we use the uniform triangular mesh with $40 \times 40 \times 2$ elements. Third and fourth order WENO schemes are used. We show the numerical results at $t = 1$ in Figure 3.15.

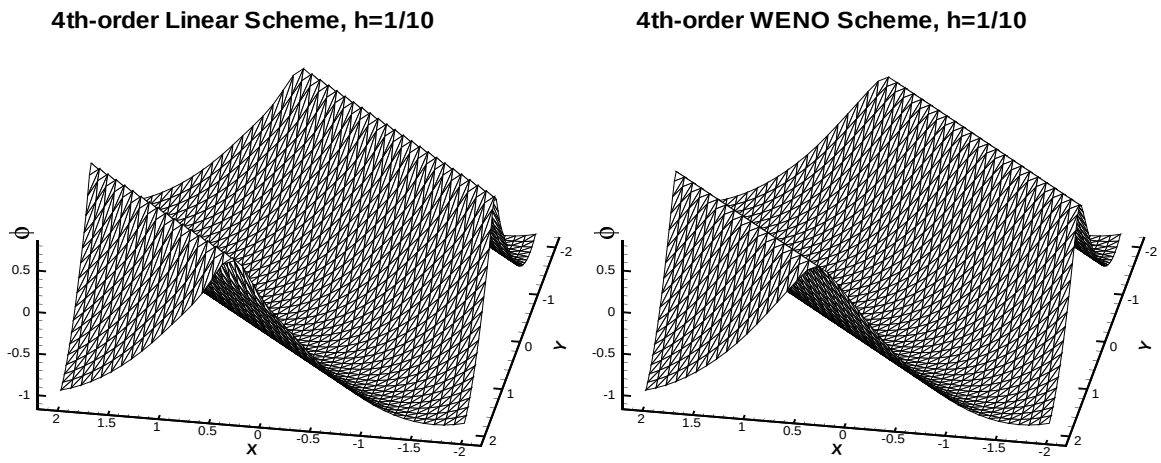

FIG. 3.8. *Two-dimensional Burgers equation,* $t = 1.5/\pi^2$. *Uniform mesh with* $h = \frac{1}{10}$. *Left: fourth order linear scheme; right: fourth order WENO scheme.*

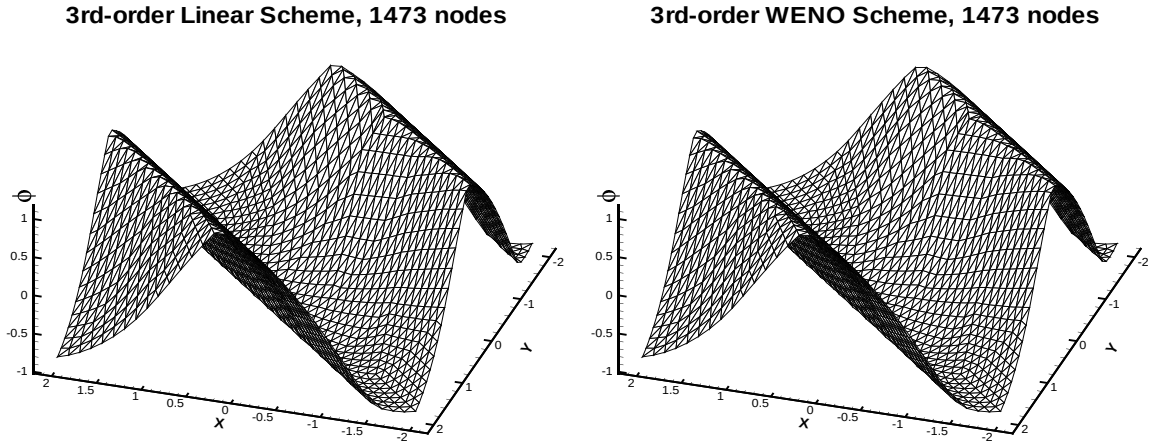

FIG. 3.9. *Two dimensions,* $H(u,v) = -\cos(u+v+1)$, $t = 1.5/\pi^2$. *Nonuniform mesh with* $N = 1473$ *nodes. Left: third order linear scheme; right: third order WENO scheme.*

Example 3.11. A problem from optimal control:

$$\begin{cases} \phi_t + (\sin y)\phi_x + (\sin x + sign(\phi_y))\phi_y - \frac{1}{2}\sin^2 y - (1-\cos x) = 0, & -\pi < x < \pi, -\pi < y < \pi, \\ \phi(x,y,0) = 0. \end{cases} \tag{3.11}$$

with periodic boundary conditions, see [13].

We use the uniform triangle mesh with $60 \times 60 \times 2$ elements. This means each of the 60×60 rectangles is cut in two triangles diagonally. Third and fourth order WENO schemes are used. The solution at $t = 1$ is shown in Figure 3.16, and the optimal control $\omega = sign(\phi_y)$ is shown in Figure 3.17.

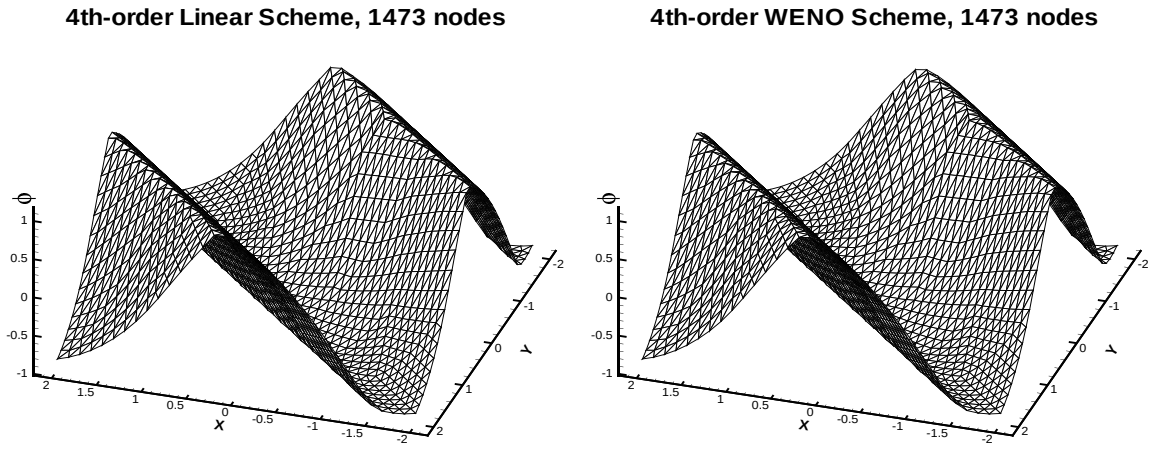

FIG. 3.10. *Two dimensions,* $H(u,v) = -\cos(u+v+1)$, $t = 1.5/\pi^2$. *Nonuniform mesh with* $N = 1473$ *nodes. Left: fourth order linear scheme; right: fourth order WENO scheme.*

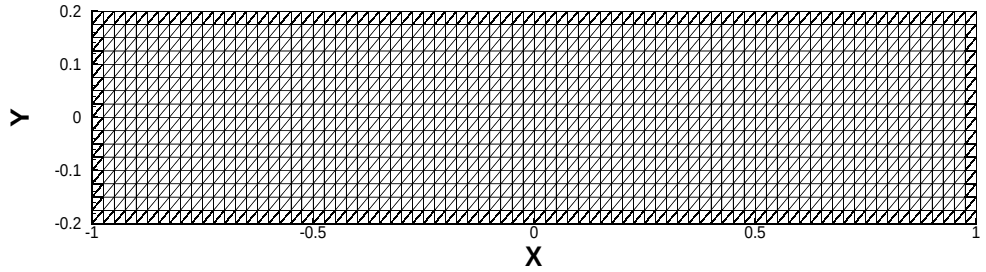

FIG. 3.11. *Mesh for Example 3.9 with* $h = \frac{1}{40}$.

Example 3.12. 2D eikonal equation with a non-convex Hamiltonian, which arises in geometric optics [9]:

$$\begin{cases} \phi_t + \sqrt{\phi_x^2 + \phi_y^2 + 1} = 0, \qquad 0 \le x < 1, 0 \le y < 1, \\ \phi(x,y,0) = 0.25(\cos(2\pi x) - 1)(\cos(2\pi y) - 1) - 1. \end{cases} \tag{3.12}$$

We use the third-order and fourth order WENO schemes. We use both the uniform meshes and the non-uniform mesh shown in Figure 3.18, and present the results in Figure 3.19 for the non-uniform mesh case at $t = 0.6$.

Example 3.13. The level set equation in a domain with a hole:

$$\begin{cases} \phi_t + sign(\phi_0)(\sqrt{\phi_x^2 + \phi_y^2} - 1) = 0, \qquad \frac{1}{2} < \sqrt{x^2 + y^2} < 1, \\ \phi(x,y,0) = \phi_0(x,y). \end{cases} \tag{3.13}$$

This problem comes from [17]. The solution ϕ to (3.13) has the same zero level set as ϕ_0, and the steady state solution is the distance function to that zero level curve. In this example, the exact steady state solution is the distance function to the inner boundary of the domain. We compute the time-dependent problem to reach a steady state solution, using the exact solution for the boundary condition. The mesh is shown in Figure 3.20, and the result using the fourth-order WENO scheme is shown in Figure 3.21.

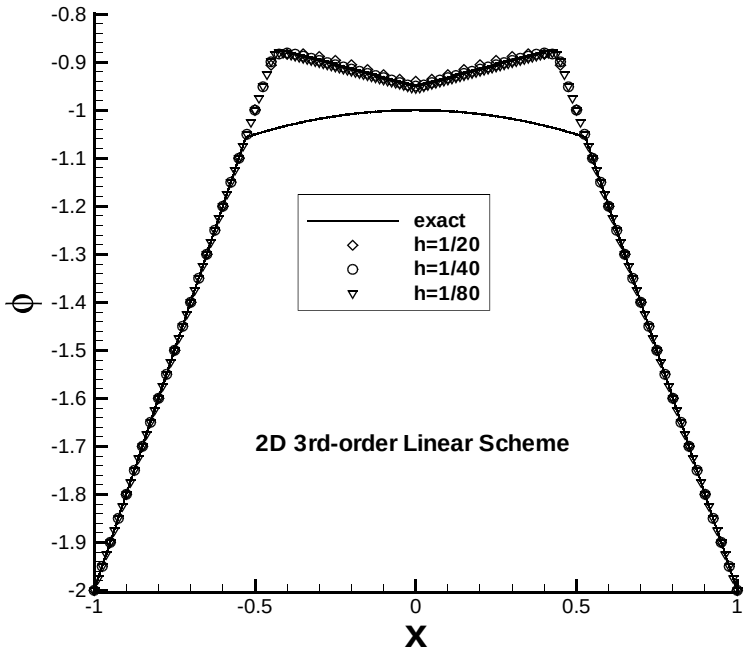

FIG. 3.12. *Convergence study for Example 3.9, linear scheme, central line cut at $y = 0$. Solid line is the exact solution; diamonds are for the coarsest mesh with $h = \frac{1}{20}$, circles are for the next refined mesh with $h = \frac{1}{40}$, and reverse triangles are for the most refined mesh with $h = \frac{1}{80}$.*

Example 3.14. The problem of a propagating surface:

$$\begin{cases} \phi_t - (1 - \varepsilon K)\sqrt{1 + \phi_x^2 + \phi_y^2} = 0, \qquad 0 \le x < 1, 0 \le y < 1, \\ \phi(x, y, 0) = 1 - \frac{1}{4}(\cos 2\pi x - 1)(\cos 2\pi y - 1), \end{cases} \tag{3.14}$$

where K is the mean curvature defined by

$$K = -\frac{\phi_{xx}(1 + \phi_y^2) - 2\phi_{xy}\phi_x\phi_y + \phi_{yy}(1 + \phi_x^2)}{(1 + \phi_x^2 + \phi_y^2)^{\frac{3}{2}}}, \tag{3.15}$$

and ε is a small constant. A periodic boundary condition is used.

This problem came from [12]. We use the fourth order WENO scheme, and the second derivative terms are approximated by those of the interpolating polynomial using the stable big stencil of our fourth order scheme discussed in section 2. For $\varepsilon = 0$ (pure convection), whose solution will develop a discontinuous derivative in the center of the domain, we use the non-uniform mesh shown in Figure 3.18, and for $\varepsilon = 0.1$, we use the uniform mesh with $50 \times 50 \times 2$ elements since the solution is smooth. The results are shown in Figure 3.22. Notice that the surfaces at $t = 0$ and $t = 0.1$ (for $\varepsilon = 0.1$) are shifted downward in order to show the detail of the solution at later time.

Example 3.15. A problem from computer vision:

$$\begin{cases} \phi_t + I(x, y)\sqrt{1 + \phi_x^2 + \phi_y^2} - 1 = 0, \qquad -1 < x < 1, -1 < y < 1, \\ \phi(x, y, 0) = 0 \end{cases} \tag{3.16}$$

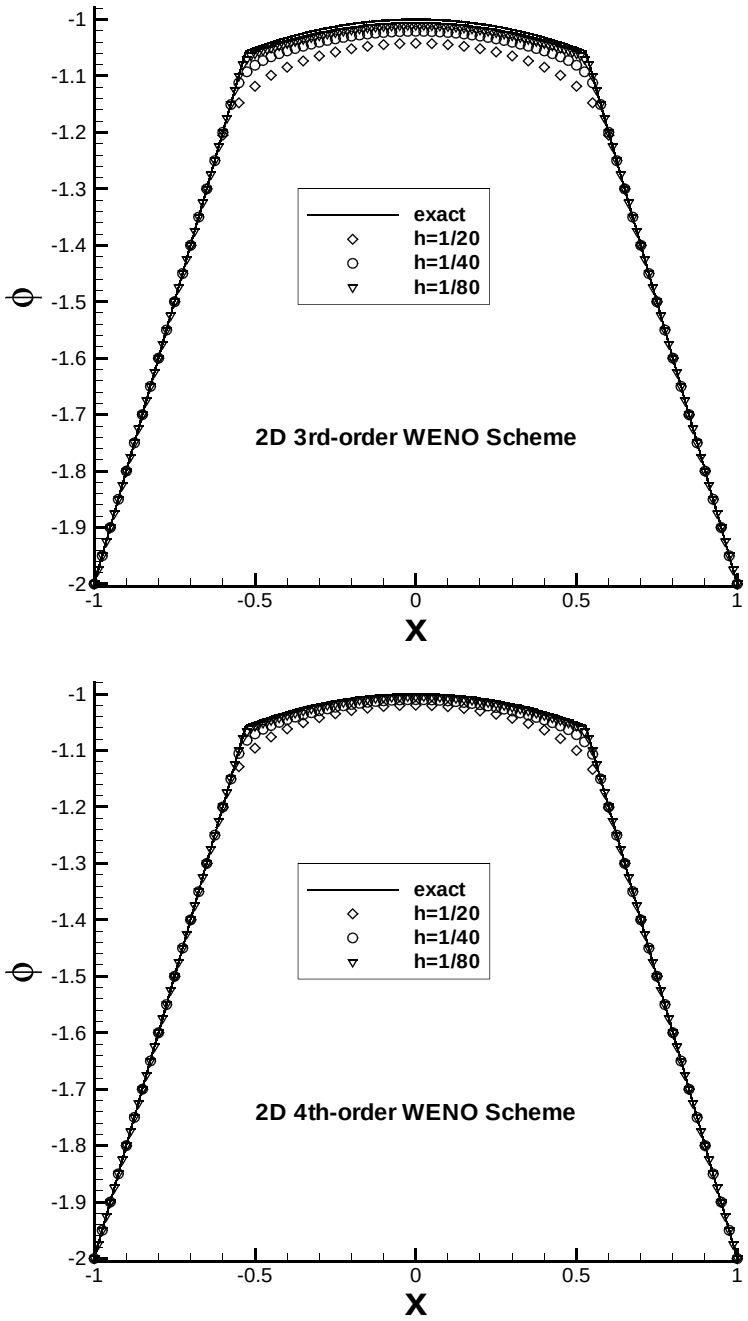

FIG. 3.13. *Convergence study for Example 3.9, WENO schemes, central line cut at $y = 0$. Solid line is the exact solution; diamonds are for the coarsest mesh with $h = \frac{1}{20}$, circles are for the next refined mesh with $h = \frac{1}{40}$, and reverse triangles are for the most refined mesh with $h = \frac{1}{80}$. Top: third order WENO schemes; bottom: fourth order WENO schemes.*

where I is defined by

$$I(x,y) = \frac{1}{\sqrt{1+(1-|x|)^2+(1-|y|)^2}} \tag{3.17}$$

with $\phi = 0$ as the boundary condition.

This problem comes from [14]. The steady state solution of this problem is the shape lighted by a source

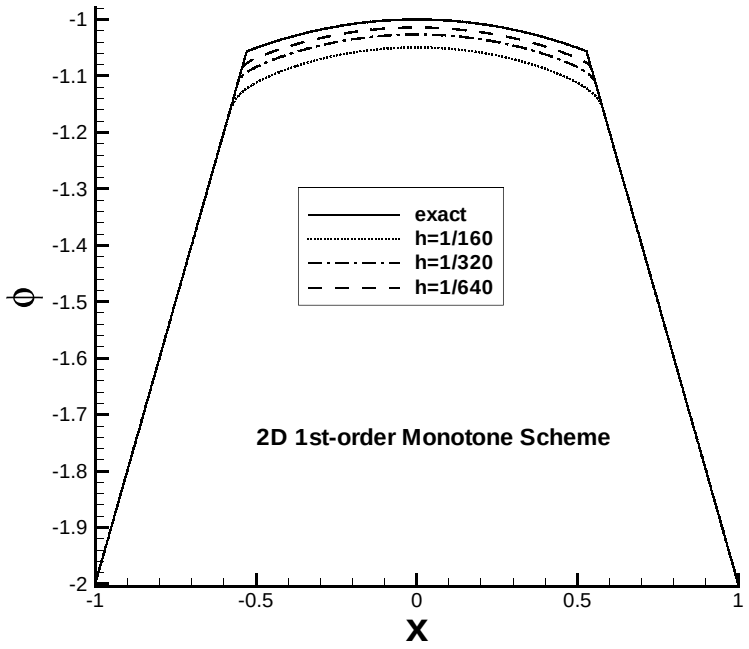

FIG. 3.14. *Convergence study for Example 3.9, first order monotone scheme, central line cut at $y = 0$. Solid line is the exact solution; dotted line is for the coarsest mesh with $h = \frac{1}{160}$, dash-dotted line is for the next refined mesh with $h = \frac{1}{320}$, and the dashed line is for the most refined mesh with $h = \frac{1}{640}$.*

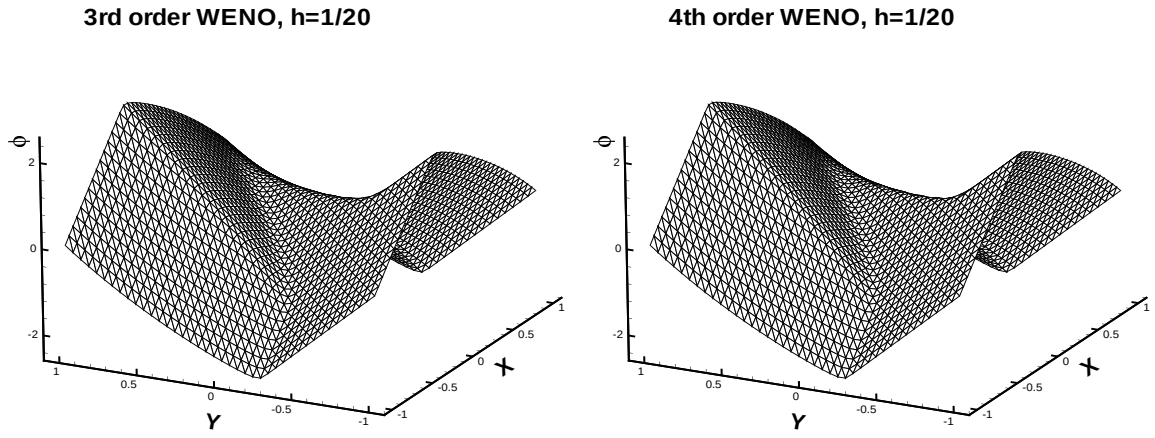

FIG. 3.15. *Two-dimensional Riemann problem, $H(u, v) = \sin(u + v)$, $t = 1$. Uniform triangle mesh with $h = \frac{1}{20}$. Left: third order WENO scheme; right: fourth order WENO scheme.*

located at infinity with vertical direction. The solution is not unique if there are points at which $I(x, y) = 1$. Conditions must be prescribed at one of those points where $I(x, y) = 1$. The exact steady solution is

$$\phi(x, y, \infty) = (1 - |x|)(1 - |y|) \tag{3.18}$$

We use the uniform mesh with $50 \times 50 \times 2$ elements and the third-order and fourth-order WENO schemes to compute the steady state solution. The boundary conditions are imposed exactly from the above exact steady solution. The results are shown in Figure 3.23.

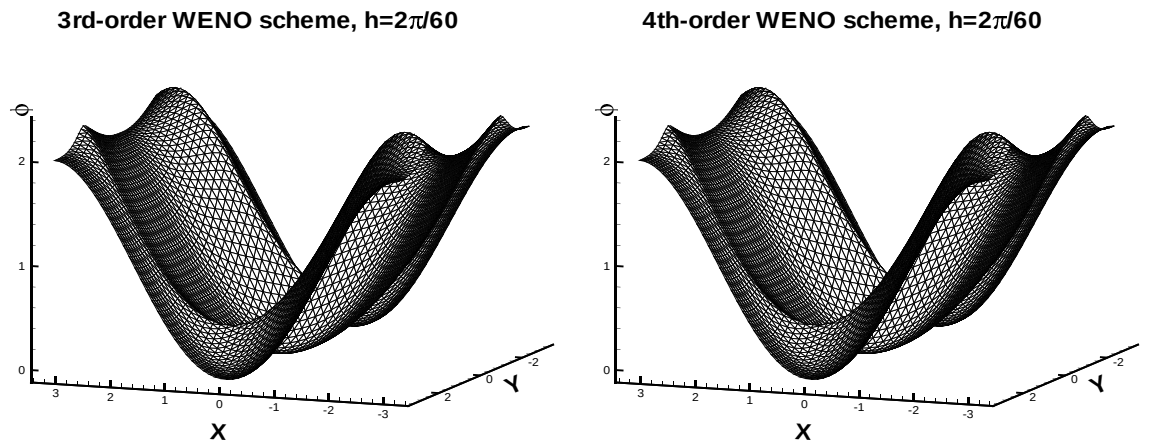

FIG. 3.16. *Control problem, $t = 1$. Uniform triangle mesh with $h = \frac{2\pi}{60}$. Left: third order WENO scheme; right: fourth order WENO scheme.*

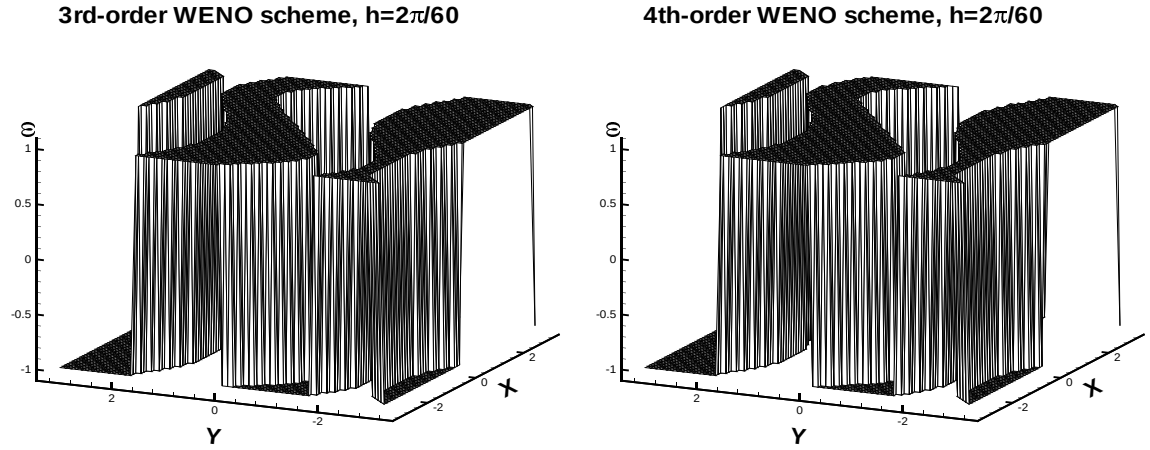

FIG. 3.17. *Control problem, $t = 1, \omega = sign(\phi_y)$. Uniform triangle mesh with $h = \frac{2\pi}{60}$. Left: third order WENO scheme; right: fourth order WENO scheme.*

4. Concluding remarks. We have constructed third order and fourth order WENO schemes for Hamilton-Jacobi equations based on 2D triangular meshes. Extensive numerical experiments have been performed to find the most robust strategies in the choice and grouping of stencils for the WENO schemes. These methods have high-order accuracy for smooth problems and high-resolution for singularity of derivatives. No parameters have to be tuned in the algorithms. Numerical Examples are shown to illustrate the capability of the method. These methods should be useful if geometry or adaptivity requires unstructured meshes for the solution of Hamilton-Jacobi equations.

REFERENCES

[1] R. ABGRALL, *Numerical discretization of the first-order Hamilton-Jacobi equation on triangular meshes*, Communications on Pure and Applied Mathematics, 49 (1996), pp. 1339–1373.

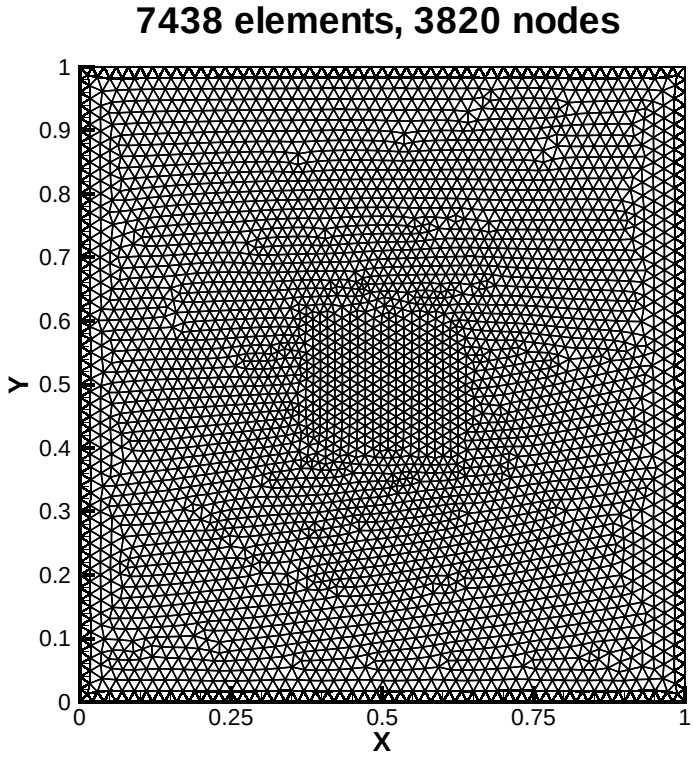

FIG. 3.18. *The Non-uniform mesh for the 2D eikonal equation.*

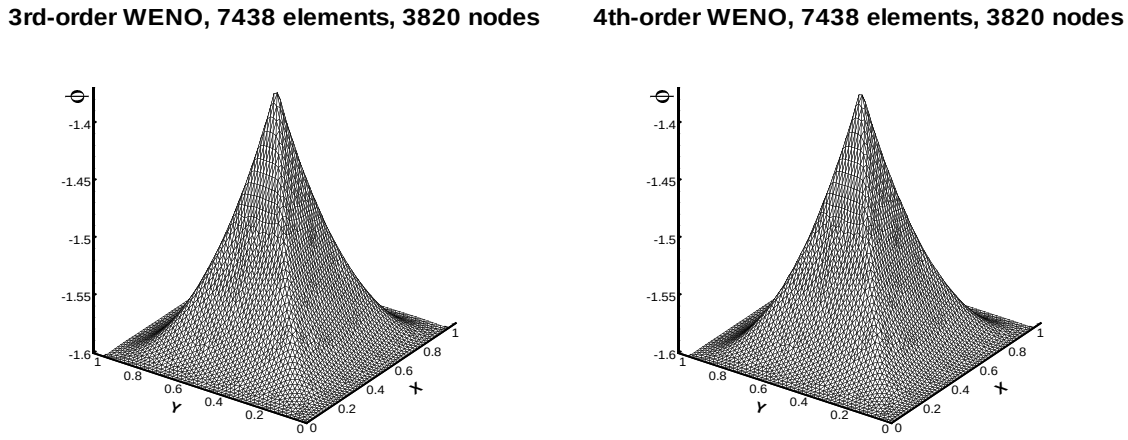

FIG. 3.19. *The 2D eikonal equation,* $H(u,v) = \sqrt{u^2+v^2+1}$, $t = 0.6$, *Non-uniform mesh. Left: third order WENO scheme; right: fourth order WENO scheme.*

[2] S. AUGOULA AND R. ABGRALL, *High order numerical discretization for Hamilton-Jacobi equations on triangular meshes*, Journal of Scientific Computing, 15 (2000), pp. 197–229.

[3] D. BALSARA AND C.-W. SHU, *Monotonicity preserving weighted essentially non-oscillatory schemes with increasingly high order of accuracy*, Journal of Computational Physics, 160 (2000), pp. 405–452.

[4] T. BARTH AND J. SETHIAN, *Numerical schemes for the Hamilton-Jacobi and level set equations on triangulated domains*, Journal of Computational Physics, 145 (1998), pp. 1–40.

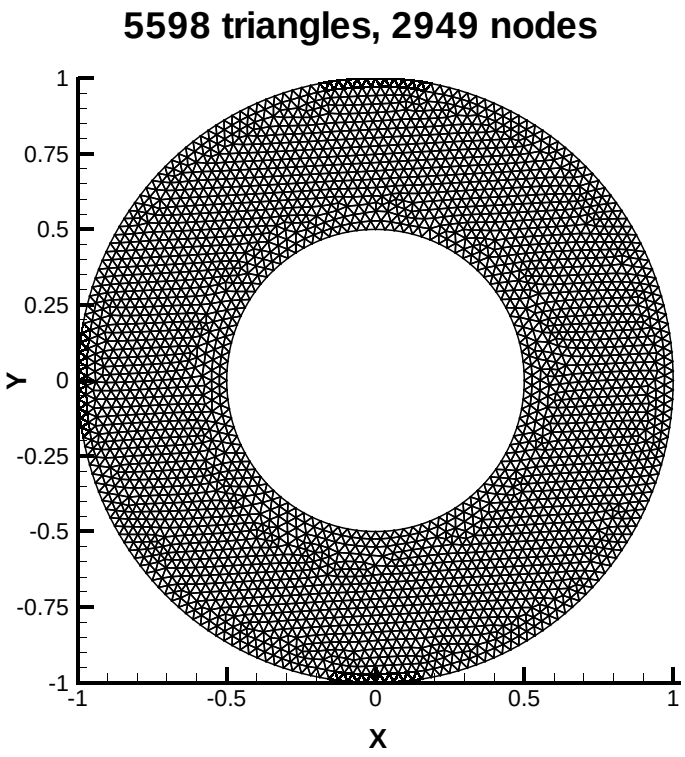

FIG. 3.20. *The mesh for level set equation.*

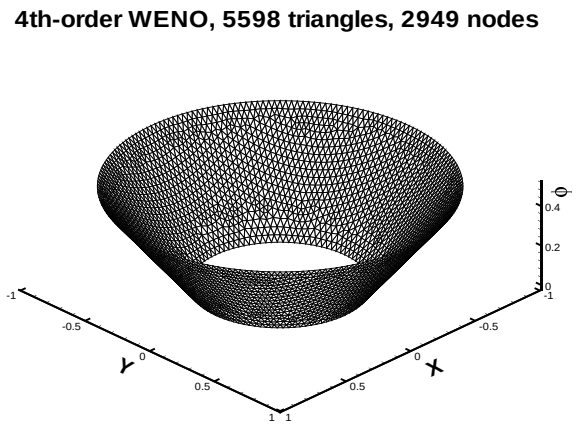

FIG. 3.21. *The steady state solution for level set equation using the fourth order WENO scheme.*

[5] C. HU AND C.-W. SHU, *A discontinuous Galerkin finite element method for Hamilton-Jacobi equations*, SIAM Journal on Scientific Computing, 21 (1999), pp. 666–690.

[6] C. HU AND C.-W. SHU, *Weighted Essentially Non-Oscillatory Schemes on Triangular Meshes*, Journal of Computational Physics, 150 (1999), pp. 97–127.

[7] G. JIANG AND D.-P. PENG, *Weighted ENO schemes for Hamilton-Jacobi equations*, SIAM Journal on Scientific Computing, 21 (2000), pp. 2126–2143.

[8] G. JIANG AND C.-W. SHU, *Efficient implementation of weighted ENO schemes*, Journal of Computa-

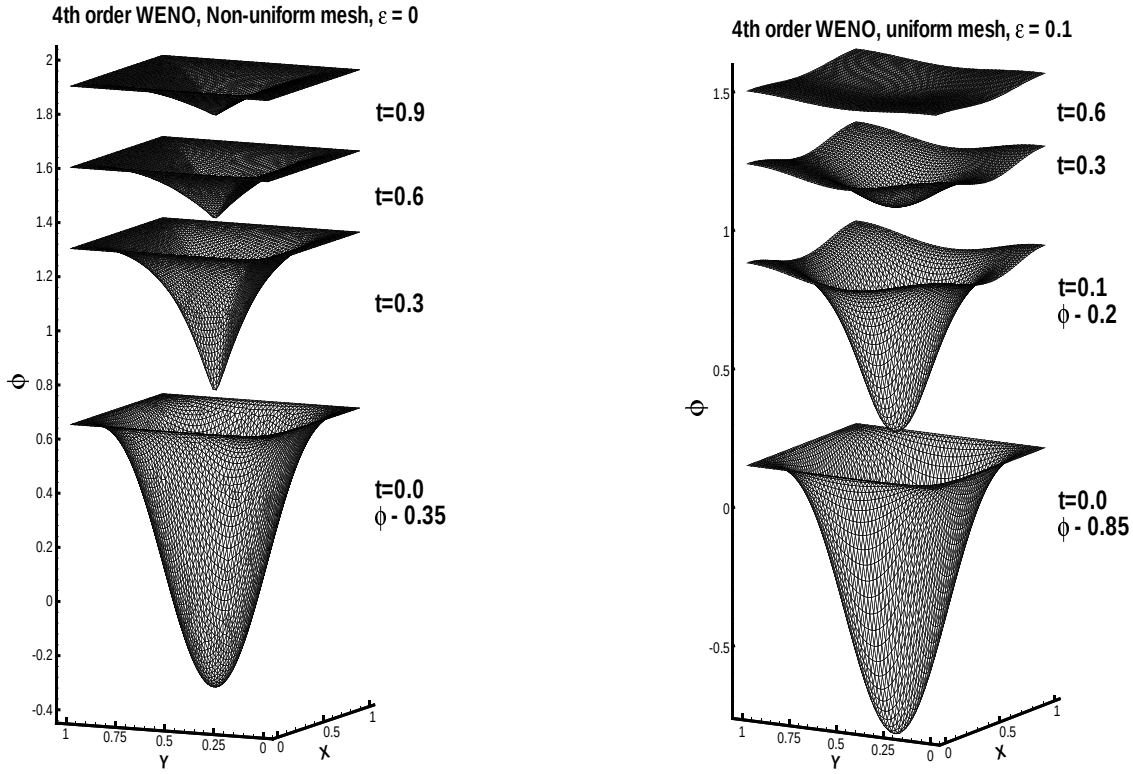

FIG. 3.22. *Propagating surfaces. Fourth order WENO. Left:* $\varepsilon = 0$ *with a non-uniform mesh; right:* $\varepsilon = 0.1$ *with a uniform mesh.*

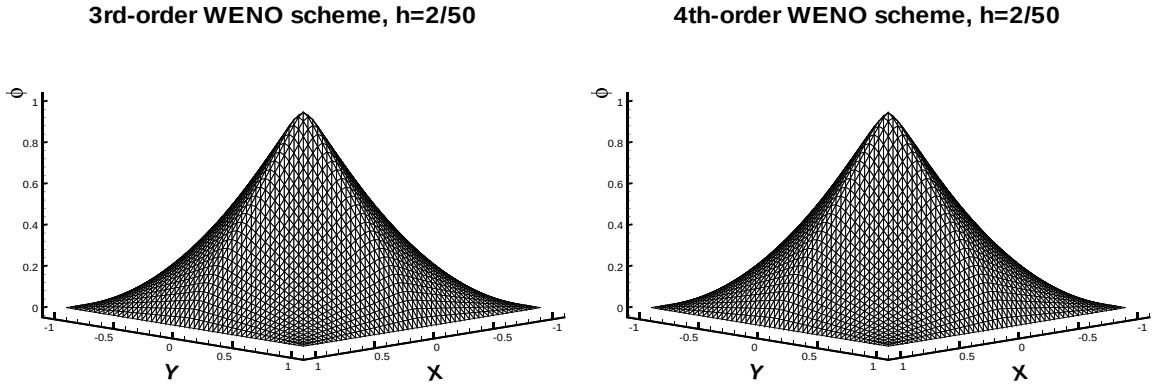

FIG. 3.23. *Computer vision problem. Left: third order WENO; right: fourth order WENO.*

tional Physics, 126 (1996), pp. 202–228.

[9] S. JIN AND Z. XIN, *Numerical passage from systems of conservation laws to Hamilton-Jacobi equations and relaxation schemes*, SIAM Journal on Numerical Analysis, 35 (1998), pp. 2385–2404.

[10] C.-T. LIN AND E. TADMOR, *High-resolution non-oscillatory central schemes for approximate Hamilton-Jacobi equations*, SIAM Journal on Scientific Computing, 21 (2000), pp. 2163–2186.

[11] X.-D. LIU, S. OSHER AND T. CHAN, *Weighted essentially non-oscillatory schemes*, Journal of Com-

putational Physics, 115 (1994), pp. 200–212.

[12] S. Osher and J. Sethian, *Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulations*, Journal of Computational Physics, 79 (1988), pp. 12–49.

[13] S. Osher and C.-W. Shu, *High-order essentially nonoscillatory schemes for Hamilton-Jacobi equations*, SIAM Journal on Numerical Analysis, 28 (1991), pp. 907–922.

[14] E. Rouy and A. Tourin, *A viscosity solutions approach to shape-from-shading*, SIAM Journal on Numerical Analysis, 29 (1992), pp. 867–884.

[15] J. Shi, C. Hu and C.-W. Shu, *A technique of treating negative weights in WENO schemes*, Journal of Computational Physics, to appear.

[16] C.-W. Shu and S. Osher, *Efficient implementation of essentially non-oscillatory shock capturing schemes*, Journal of Computational Physics, 77 (1988), pp. 439–471.

[17] M. Sussman, P. Smereka and S. Osher, *A level set approach for computing solution to incompressible two-phase flow*, Journal of Computational Physics, 114 (1994), pp. 146–159.